SISTEMA DIFERENCIAL DE RASTREAMENTO DE MOVIMENTOS ANGULARES – DSAMT

Rodrigo de Souza Santos
Universidade Estácio de Sá – UNESA, Campus Sulacap, Rio de Janeiro, Brasil
rodrigodesouzas01@gmail.com

Paulo Cezar Rocha Silveira
Universidade Federal do Rio de Janeiro - PEN/COPPE/UFRJ, Centro de Tecnologia, Rio de Janeiro, Brasil
paulo@imp.ufrj.br

RESUMO

Neste artigo será apresentada a proposta de utilização de Sistemas Micro Eletro Mecânicos (MEMS), para o desenvolvimento de um Sistema Diferencial de Rastreamento do Movimento Angular (SDRMA), com a finalidade de aumentar a percepção do operador de um Veículo Remotamente Pilotado (RPV). A proposta é de utilizar Unidades de Medida Inercial (IMU) para o desenvolvimento de um sistema que terá como função monitorar os movimentos da cabeça do operador de um RPV e, através destes movimentos, controlar uma câmera de vídeo instalada no veículo fazendo com que a linha de visão da câmera acompanhe estes movimentos, proporcionando ao operador um monitoramento mais eficiente do cenário explorado pelo RPV.


DIFFERENTIAL SYSTEM OF ANGULAR MOVEMENT TRACKING – DSAMT

ABSTRACT

This article presents the proposal to use Micro Electro Mechanical Systems (MEMS) for the development of a Differential Angular Movement Tracking System (DSAMT), in order to increase the perception of the operator of a Remotely Pilot Vehicle (RPV). The proposal is to use Units of Inertial Measure (IMU) to develop a system that will monitor the movements of the head of the operator of a RPV and, through these movements, control a video camera installed in the vehicle, making the line of vision of the camera accompany these movements, giving the operator more efficient monitoring of the scenario explored by RPV.

Keywords: Micro Electro Mechanical System (MEMS). Inertial Measurement Unit (IMU). Remotely Piloted Vehicle (RPV).
1 INTRODUCTION

Controlling a device at a distance or making it work without human intervention is a long-sought goal. At the present time, these devices are present in the everyday of humanity, from toys to sophisticated systems of remote operation or autonomous operation.

One system that has had great relevance is the RPV (Remotely Piloted Vehicle). These vehicles have been used in activities involving risks to human health or in places of difficult access. As an example of these vehicles, we can mention the Cobham Teodor Antibomb Robot used to disarm bombs (CASTELLI, 2013) or the T-Hawk drone used by the Japanese to access places with high radiation as a result of the Fukushima disaster (GARRETT, 2013).

Another widely used RPV is the ROV (Remotely Operated Vehicle) applied in operations in ultra-deep waters where high pressure at the seabed makes it impossible to be used by humans for these operations, the use of these vehicles made possible the exploration of oil in the Brazilian pre-salt (INSTITUTO HIDROGRÁFICO, 2016).

The information transmitted to the operators of these vehicles is crucial for making decisions and consecutive safe operation in the most diverse environments. The sensors embedded in these vehicles detect this information, which is then transmitted to the remote station. The greater the quantity and quality of these different pieces of information, the better the operator’s decision making.

One system used to obtain this information is the video capture or capture system. Cameras are installed in the vehicles and the video information transmitted to the remote station. These cameras fixed at certain points of the vehicles capture images in the opening area of their lenses. An existing system currently uses a camera installed on the front of a drone; this system is called First Person View (FPV).

A drone with FPV has to transmit images from its camera to the user who then sends control signals back. Although some of the stimuli associated with aircraft piloting, such as inertial effects on the pilot and their own sense of balance, are lacking, the FPV can provide an experience very similar to that of a real flight and is an effective way to control a drone already that the pilot acquires a reasonable level of ability (DOUGHERTY, 2019).

Another system that uses imaging is the Raytheon AN / AAS-52 system installed in the military drone MQ-9 Reaper. The Raytheon NA / AAS-52 is a powerful target
acquisition system that includes a color television camera, also with infrared, and another TV with the laser system to designate targets, that is, the control station itself (which can be in Florida), the operator assigns the target and releases the missiles and/or bombs with the drone being 14,000 km away in Afghanistan, for example (CANALTECH, 2019).

In this context, it is interesting to develop an auxiliary system, which allows enlarging the angle of view of these cameras, increasing the visual amplitude of the information generated by the video capture system.

This article describes the development of a system that allows the remote control of the angular movements of the axes of a video camera installed in a RPV by monitoring the movements of the head of the vehicle operator.

2 SYSTEM DESCRIPTIONS

In order to monitor the angular movements of the head of an RPV operator, the data produced by an IMU (Inertial Measurement Unit) in the MPU-6050 sensor housing will be used. The IMU used in this project consists of two different types of MEMS sensors.

The MEMS MPU-6050 device from InvenSense brings together an accelerometer and a gyroscope in the same housing. Each sensor is able to monitor the variations of the three axes (X, Y and Z) individually and simultaneously, which gives the set the six degree freedom of reading also known as 6DOF (Degrees Of Freedom). For a more accurate reading, the MPU-6050 has a digital motion processor (DMP) which uses filters and algorithms that treat and match the signals generated by the sensors, transforming this information into inertial measurements. By having these characteristics the device has received the designation of the prefix MPU which means movement processor unit (INVENSENSE INC., 2013).

The information provided by the DMP of the MPU-6050 sensor is sufficient to allow the control of the output device (camera), but the operator would have mobility and locomotion restrictions because the sensor would perceive the movements made by the operator and not only the movements of his head. To solve this problem it was proposed to use a second sensor attached to the operator’s body in order to monitor the movements produced by the operator’s body so that they could be neglected by the system. In this way, angular changes produced by displacements would be perceived by the two sensors in a reciprocal way, in contrast, head movements would be perceived only by the sensor attached to the head, therefore it is possible to determine the specific
movements of the operator’s head through the subtraction (difference) of the values provided by both sensors.

For the development of DSAMT, a prototype was built to demonstrate a video camera being controlled by DSAMT. The prototype is composed of two units, a transmission unit and a receiving unit. The transmission unit is responsible for acquiring the angular information of the sensors, processing and filtering this information and then transmitting it to the receiving unit. The receiving unit, which has the information transmitted, will be responsible for controlling camera movements. For the assembly of the prototype the following systems are used:

### Schedule 1 - List of materials

<table>
<thead>
<tr>
<th>Description</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino Nano with Atmega 328P Microcontroller</td>
<td>01 Unit</td>
</tr>
<tr>
<td>Arduino Uno with Atmega 328 Microcontroller</td>
<td>01 Unit</td>
</tr>
<tr>
<td>2.4GHz XBee-PRO RF Module</td>
<td>02 Unit</td>
</tr>
<tr>
<td>Sensor 6 DOF MPU-6050</td>
<td>02 Unit</td>
</tr>
<tr>
<td>Power distribution board</td>
<td>01 Unit</td>
</tr>
<tr>
<td>Xbee Shield for Arduino</td>
<td>01 Unit</td>
</tr>
<tr>
<td>Virtual reality glasses VR 3d BOX</td>
<td>01 Unit</td>
</tr>
<tr>
<td>4S LiPo Battery</td>
<td>01 Unit</td>
</tr>
</tbody>
</table>

Source: (the authors)

The two MPU-6050 sensors are installed on the transmitter unit. One of the sensors is affixed to a VR-type spectacles, it is through these glasses that the system perceives the movement produced by the operator’s head. The second sensor is installed in a plastic box where the other systems integrating the unit are packed. This box is affixed to the body of the operator in order to perceive the movement produced by the displacement of your body.

To perform the acquisition and processing of the data generated by the sensors an Arduino Nano module is employed. This module is responsible for all processing and coding of the variables for later transmission of this data to the receiving unit. The transmission of this data is done through the RF XBee module enabling wireless communication between the transmitting and receiving units. The transmitter unit is powered by two 9 Vdc batteries. Figure 01 shows the Transmission Unit (TX) installed on the operator’s chest and Figure 02 shows the parts of the transmission unit.
An XBee RF module, an Arduino UNO, an XBee Shield, a power distribution board, a TS832 video transmitter module, a 4S LiPo battery and a camera installed on a stand are installed in the receiver unit (RX). Servo motors, Y (incline) and Z (turn), produce the movements on the respective axes. The XBee module is responsible for receiving data sent from the Transmitter Unit and forwarding them to the Arduino Uno. The Arduino
Uno receives these pieces of information via serial port and processes them with the purpose of separating them, identifying them and using them to control the servos of the camera holder. The unit system is powered by a 4S LiPo battery, this power is regulated and distributed by a distribution board. Figure 03 shows the receiving unit and Figure 04 shows the systems installed in the unit.

Figure 3 - Receiving Unit (RX)

Source: the authors

Figure 4 - Components of the Receiving Unit (RX)

Source: the authors
Figure 05 shows the summary of the flow of operation of the transmitting unit (TX).

![Figure 5 - Transmitter TX unit flow](image)

Source: the authors.

Figure 06 shows the summary of the operating flow of the receiving unit (RX).

![Figure 6 - Receiver unit flow (RX)](image)

Source: The authors.

The programming routines were written in the C language and are presented below. Only the routines related to the main processing are presented below.

### 2.1 Main programming routine of the transmitter unit (TX)

```c
// === MAIN PROGRAM LOOP ===
void loop() {

//==========================================
//== SENSOR A ==============================
//==========================================
```

//Seleção do Sensor
digitalWrite(AD0A_PIN,LOW);
analogWrite(AD0B_PIN,ADD_ON);

// if programming failed, don’t try to do anything
if (!dmpReady) return;
// reset interrupt flag and get INT_STATUS byte
mpuInterrupt = false;
mpuIntStatus = mpu.getIntStatus();

// get current FIFO count
fifoCount = mpu.getFIFOCount();

// check for overflow (this should never happen unless our code is too inefficient)
if ((mpuIntStatus & 0x10) || fifoCount == 1024) {
  // reset so we can continue cleanly
  mpu.resetFIFO();
}

// otherwise, check for DMP data ready interrupt (this should happen frequently)
} else if (mpuIntStatus & 0x02) {
  // wait for correct available data length, should be a VERY short wait
  while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();

  // read a packet from FIFO
  mpu.getFIFOBytes(fifoBuffer, packetSize);

  // track FIFO count here in case there is > 1 packet available
  // (this lets us immediately read more without waiting for an interrupt)
  fifoCount -= packetSize;

  // display quaternion values in easy matrix form: w x y z
  mpu.dmpGetQuaternion(&q, fifoBuffer);

  //Cálculo dos ângulos em graus baseado nos valores do quaternion
  vetorXA=calculo(q.x);
  vetorYA=calculo(q.y);
  vetorZA=calculo(q.z);
}

//=----------------------------------------------------------------------------------
// Selecção do Sensor
digitalWrite(AD0B_PIN,LOW);
analogWrite(AD0A_PIN,ADD_ON);

// if programming failed, don’t try to do anything
if (!dmpReady) return;

// reset interrupt flag and get INT_STATUS byte
mpuInterrupt = false;
mpuIntStatus = mpu.getIntStatus();

// get current FIFO count
fifoCount = mpu.getFIFOCount();

// check for overflow (this should never happen unless our code is too inefficient)
if ((mpuIntStatus & 0x10) || fifoCount == 1024) {
    // reset so we can continue cleanly
    mpu.resetFIFO();
}

// otherwise, check for DMP data ready interrupt (this should happen frequently)
else if (mpuIntStatus & 0x02) {
    // wait for correct available data length, should be a VERY short wait
    while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();

    // read a packet from FIFO
    mpu.getFIFOBytes(fifoBuffer, packetSize);

    // track FIFO count here in case there is > 1 packet available
    // (this lets us immediately read more without waiting for an interrupt)
    fifoCount -= packetSize;

    // display quaternion values in easy matrix form: w x y z
    mpu.dmpGetQuaternion(&q, fifoBuffer);

    // Cálculo dos ângulos em graus baseado nos valores do quaternion
    vetorXB=calculo(q.x);
    vetorYB=calculo(q.y);
    vetorZB=calculo(q.z);
}
// Angular Differences
difX = calcDiff(vetorXA, vetorXB);
difY = calcDiff(vetorYA, vetorYB);
difZ = calcDiff(vetorZA, vetorZB);

2.2 Receiver unit program (RX)
// Biblioteca para aplicação no receptor do Sistema Diferencial de Rastreamento de Movimento Angular

// #include “Wire.h”
#include <Servo.h>
#define POS_INI 90 // Posição inicial dos servos
// Declaração dos Servos
Servo servoX;
Servo servoY;
Servo servoZ;

// Variáveis
float inSerial;
float anguloX, anguloY, anguloZ;

// Setup
void setup()
{
    servoX.attach(9);
    servoY.attach(10);
    servoZ.attach(11);

    // Posicionamento inicial dos servos
    servoX.write(POS_INI);
    servoY.write(POS_INI);
    servoZ.write(POS_INI);

    Serial.begin(57600);
    // while (!Serial); // wait for Leonardo enumeration, others continue immediately
}

void loop()
{
    if (Serial.available() > 1)  // Avaliando o conteúdo da Serial
    {
        inSerial = Serial.parseFloat();
    }
if ((inSerial >= 1000)&&(inSerial < 2000))
{
    anguloX = inSerial-1000;
    servoX.write(anguloX);
}
else if ((inSerial >= 2000)&&(inSerial < 3000))
{
    anguloY = inSerial-2000;
    servoY.write(anguloY);
}
else if ((inSerial >= 3000)&&(inSerial < 4000))
{
    anguloZ = inSerial-3000;
    servoZ.write(anguloZ);
}

Other routines were used to treat the data as filters for delay in the variations of values and coding algorithms for transmission of values between TX and RX units.

3 RESULTS

The results demonstrated the efficiency of the proposed system in tracking and transmitting the head movements of the RPV operator and in detecting the movements of his body in order to inhibit the interference of these movements in the camcorder control.

Below are some images in order to demonstrate the capture of images capture using the proposed system.
Figure 7 – Image capture without using the system.

Source: the authors.

Figure 8 – Image capture using the system (Front view).

Source: the authors.

Figure 9 – Image capture using the system (Right view).

Source: the authors.
4 CONCLUSIONS

The DSAMT provided a greater viewing angle of the environment allowing a better domain in controlling the movement of on RPV. The operator has freedom of viewing with the camera mounted on the stand where the movement is controlled by
the operator, ie if there is a need to change the position of the camera, it is enough for the operator to make the necessary head movements that the camera will follow these movements. This makes RPV operation safer and more reliable.

The system is still under development with the aim of improving stability and enabling the image captured by the camcorder to be projected on a screen placed in front of the operator’s eyes making it possible to make a complete, compact and portable system.

REFERENCES


