MOTION-SENSED RTOS-BASED APPLICATION CONTROL USING IMAGE PROCESSING

Manav Jaiswal¹, Akshay Gavandi², Kundan Srivastav³, Dr. Srija Unnikrishnan⁴

¹,²,³UG students, Department of Electronics Engineering, Fr. Conceicao Rodrigues College of Engineering
⁴Professor and Principal, Fr. Conceicao Rodrigues College of Engineering Fr. Agnel Ashram, Bandstand, Bandra (W), Mumbai, Maharashtra, India. Pin Code: 400 050

ABSTRACT

The day-by-day growing Human Computer Interaction (HCI) technology has greatly impacted the existing electronic systems allowing more easy and interactive accessing methods. This paper elaborates a way of access and control of computer applications and embedded systems by motion sensing using color of objects. For materializing the concept, a small tank game, based on a microcontroller running on a Real-Time Operating System (RTOS), has been created which acts as an application to be controlled. Micrium’s µc/OS II is used as the RTOS running on uNiboard development board, with Atmel’s ATmega64 microcontroller. Image processing for sensing the motion of the colored object is done using MATLAB.

Controlling the game using the above mentioned way is just one application and this system can be used to simplify access to digital systems and to even make them autonomous.

Keywords: Human Computer Interaction, Image Processing, MATLAB, Micrium’s µc/OS II.

1. INTRODUCTION

With advancement in technology, significant changes have been observed in accessing techniques of electronic devices. During electronics industry’s infancy, electronic devices used to be controlled by on-device switches and buttons. With advancing technology, it has been gradually replaced by remote controls. With the advent of new image processing and fast computing technologies, the human imagination of controlling electronic devices by motion sensing, color recognition and object recognition has been materialized.

Presently, there are many algorithms present for motion detection. Some of them include Kalman filter algorithm, Camshift algorithm, Meanshift algorithm, etc. These algorithms involve
complex processing techniques at pixel level. The challenge addressed in this paper is to propose an
easier technique for sensing motion of an object based on its color, using MATLAB, which can be
used to control a range of computer applications and embedded system devices. The employed
technique is such that it can be incorporated by anyone, for controlling their respective systems, with
basic knowledge of Image Processing Toolbox in MATLAB and a low quality camera. The
implementation is easy and cheap because the components required are just a low quality camera and
MATLAB software with Image Processing Toolbox.

In this implementation, an input video is provided by the webcam to MATLAB, depicting
motion of a red object. Red color is used for object detection in this paper. The motion of the red
object is processed by MATLAB to control the application. An RTOS based, simple to comprehend,
tank game has been created as an application to be controlled. The Micrium’s µc/OS II is used as the
RTOS to run the game. This RTOS has been loaded on uNiboard v1.1 which is an ATMega64
microcontroller-based development board. The game is displayed, on the computer running
MATLAB, using the serial terminal software PUTTY. An impaired webcam is used, to prove the
compatibility of the recognition system with very low quality inputs.

![Figure 1: Block Diagram](image)

The paper’s structure is as follows:
- Section 2: Component Description
- Section 3: MATLAB Processing
- Section 4: RTOS-based Application
- Section 5: Control of the Game and Test Results

2. COMPONENT DISCRIPTION

The hardware and software used are:

- uNiboard V 1.1
- A computer
- MATLAB with Image Processing Toolbox, on the computer
- Micrium’s µc/OS II running on uNiboard V1.1
- An impaired webcam, giving a blurred output video, connected to the computer
- Serial terminal software PUTTY
2.1. **WEBCAM**

Features:
- VGA quality
- Low resolution
- Impaired condition, blurred output video
- Provides video to MATLAB

2.2. **uNiboard V1.1**

![uNiboard V1.1](image)

**Figure 2: uNiboard V1.1**

Features:
- Atmega64 microcontroller-based development board
- 64 KB of flash memory and 4 KB of EEPROM
- Onboard joystick, 2 physical serial ports, UART DB9 connectors, LED and LCD interfaces
- SD/MMC card interface

3. **MATLAB PROCESSING**

3.1 **OBJECT INITIALIZATION**

The following steps explain the initialization of connections with uNiboard and webcam in MATLAB:
- Create an object of webcam using function “videoinput()”
- Create a serial object for connection with uNiboard
- Define output buffer size for the serial object
- Start the video input by function “start()”

3.2 **ACQUIRING FRAME AND PROCESSING ALGORITHM**

The following flow chart explains the processing, on frame acquired from input video, to get the coordinates of centroid of the red object in the frame which is used to control the game:

![Frame processing algorithm flowchart](image)

**Figure 3: Frame processing algorithm flowchart**
As observed from the flowchart, a frame is acquired from video input of webcam by MATLAB, which is then converted to Grayscale from RGB. The resultant frame is then subtracted from red component of original frame. This step isolates the red components in the frame. Thus, the red components of the frame get highlighted. The 2-dimensional median filtering is done with a mask of order 3x3 over the resulting frame. This process is carried out using the available function:

```matlab
medfilt2(image_pointer, [3,3])
```

The frame is then converted to binary form and any object with diameter less than 300 pixels are removed. The object in the frame is then labeled and a bounding box is drawn using:

```matlab
bw = bwlabel(image_pointer, 8)
stats = regionprops(bw, 'BoundingBox', 'Centroid')
```

The coordinates of the centroid are obtained from the variable `stats` using this code:

```matlab
bc = stats(object).Centroid
n1=round(bc(1))
n2=round(bc(2))
```

The final frame from the above processing is assumed to be divided in six rectangular sections as shown in fig. 6. The variables `n1` and `n2` possess the coordinates of the centroid, as shown in Fig. 7, which are used to determine location of the red object among the six sections. The coordinates of top left corner is taken to be (0, 0) and bottom right corner to be (255, 255).

```
Figure 4: Stepwise processing of frame
```

```
Figure 5: Sectioning of frame
```
The sections of the frame, as shown in Fig. 6, are then numbered 1 to 6 from left to right and top to bottom fashion. The section size decides the sensitivity of control of the tank to the amount of motion the colored object does. Deciding the limits of the sections depends on the application to be controlled and the amount of movement of colored object, red object for application discussed, to which the application responds. The section containing the centroid of the object is determined by comparing the coordinates of centroid and section limits. The section number is passed to uNiboard using the serial handler of the established serial communication.

3.3 ALGORITHM LOOPING FOR CONTINUOUS CONTROL

The process mentioned in the section 3.2, is an algorithm to process just a single frame and the centroid coordinates obtained are of one instance only. To continuously control the game, until any response to stop the process is received, the above mentioned process in 3.2 is kept in a loop. The loop breaks only if MATLAB receives any response from uNiboard to stop sensing.

4. RTOS-BASED APPLICATION

As mentioned earlier, the game consists of a tank shooting at arbitrarily generated target point on screen. The target appears at different locations on the screen for a fixed period of time. The tank is made to shoot the target by the user. Thus, the tank is the item to be controlled by moving a red object before the webcam.

4.1 FEATURES OF MICRIUM’s µc/OS II

Following are some important features of MICRIUM’s µc/OS II RTOS, which are used for building the game:

- Task creation for multitasking.
- Mutex: It is a binary semaphore with priority inheritance mechanism for implementing synchronization between tasks.
- Mail box: It is used to post a parameter through tasks.

4.2 GAME INITIALIZATION

The following are the initialization steps for the RTOS and serial display on PUTTY:

- Initialize the operating system to start multitasking operations
- The following are the tasks used for building the game:
  - Tank movement: The motion of the tank is handled by this task.
  - Projectile movement: The motion of the bullet shot by the tank is handled by this task.
  - Target generation: This task generates the target at arbitrary locations after a specific period of time.
  - Score update: This task updates the score.
• Initialize mutex and mail box
  o Mutex acts like a token, where the task that holds the token can execute. At the end of the code the token is released so that it can be held by another task.
  o Mail box is used for communicating between tasks. A pointer sized variable is sent through mail box functions to other tasks.
• A video terminal library is created in C language for colorful graphical display on serial terminal PUTTY.
• This library is used in programming the graphics of the game as shown in Fig. 8. The initialization is done on uNiboard in following order:
  o The screen is set up with blue background and a black border.
  o A tank is constructed using a set of characters arranged to form a pattern representing a tank. The starting position of the tank is initialized.
  o The name of the game and initial score are displayed at the top of the screen.

![Figure 7: Screenshot of PUTTY running the game](image)

5. CONTROL OF THE GAME AND TEST RESULTS

After processing a frame of an instance, a variable containing section number (from 1 to 6) is passed to uNiboard which allows the OS tasks to decide the motion of the tank and shooting of the bullets. Following are the section numbers and the movement of tank if the centroid of the red object falls in that corresponding section:
1. Moves left and shoot.
2. Only shoot.
3. Moves right and shoot.
4. Move left.
5. Stay at current position.
6. Move right.

Following are the test results of processing of frames and respective screenshots of outputs. The Section number and centroid coordinates are also mentioned.

![Figure 8: Test 1 [Section 1, (74, 69)]](image)
Figure 9: Test 1 Output

Figure 10: Test 2 [Section 2, (175, 56)]

Figure 11: Test 2 Output

Figure 12: Test 3 [Section 3, (263, 45)]
Figure 13: Test 3 Output

Figure 14: Test 4 [Section 4, (51,201)]

Figure 15: Test 4 Output

Figure 16: Test 5 [Section 5, (140, 208)]
6. CONCLUSION AND FUTURE DEVELOPMENT SCOPE

The motion sensing algorithm has been successfully implemented using MATLAB and the RTOS-based application. It can be used to control a system by motion of an object where the controlling parameters are few. RTOS-based application runs on Micrium µC/OS-II and consisted of a tank game. This is a specific application developed for testing the motion-sensing algorithm and any such application can be developed as per the need. The advantage of the proposed idea is its simplicity. It can be implemented without much effort and requires inexpensive and simple configuration. Further on, hue and saturation processing can be added in the algorithm for more precise output.
7. REFERENCES

[1] Labrosse Jean J., µC/OS-II: The Real-Time Kernel

AUTHORS

Manav Jaiswal is pursuing degree of ‘Bachelor of Engineering’ in Electronics Engineering from Fr. Conceicao Rodrigues College of Engineering, Mumbai – 400050, affiliated to University of Mumbai. His current research interests include ‘Embedded Systems’, ‘Computer Vision’ and ‘Human-Computer Interaction’.

Akshay Gavandi is pursuing degree of ‘Bachelor of Engineering’ in Electronics Engineering from Fr. Conceicao Rodrigues College of Engineering, Mumbai – 400050, affiliated to University of Mumbai. His current research interests include ‘Embedded Systems’, ‘Computer Vision’ and ‘Automotive Electronics’.

Kundan Srivastav is pursuing degree of ‘Bachelor of Engineering’ in Electronics Engineering from Fr. Conceicao Rodrigues College of Engineering, Mumbai – 400050, affiliated to University of Mumbai. His current research interests include ‘Embedded Systems’, ‘Power Electronics’ and ‘Avionics’.

Dr. Srija Unnikrishnan holds a Master of Engineering degree in Electronics & Telecommunication. Furthermore, she has a Ph.D (Tech.) in Electrical Engineering. She specializes in ‘Telecommunication’ and ‘Signal Processing’. She has a teaching experience of 25 years and is currently the Principal of Fr. Conceicao Rodrigues College of Engineering.