ROBotic Open-architecture Technology for Cognition, Understanding and Behavior

Project no. 004370

RobotCub

Development of a cognitive humanoid cub

Instrument: Integrated Project
Thematic Priority: IST – Cognitive Systems

RC_IST_110_D_000_07_EXPRESSIONS_CONTROL

Expressions System Control for the iCub Head

Internal Report

Due Date:
Submission date: 8/3/2007

Start date of project: 01/09/2004
Duration: 60 months

Organisation name of lead contractor for this deliverable: IST
Responsible Person: José Santos-Victor
Authors: Alexandre Bernardino, Ricardo Nunes, Luis Vargas, Ricardo Beira, Manuel Lopes and José Santos-Victor, Instituto Superior Técnico (IST)

Revision: rev.no. 07

<table>
<thead>
<tr>
<th>Dissemination Level</th>
<th>Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PU</td>
<td>Public</td>
</tr>
<tr>
<td>PP</td>
<td>Restricted to other programme participants (including the Commission Service)</td>
</tr>
<tr>
<td>RE</td>
<td>Restricted to a group specified by the consortium (including the Commission Service)</td>
</tr>
<tr>
<td>CO</td>
<td>Confidential, only for members of the consortium (including the Commission Service)</td>
</tr>
</tbody>
</table>
# Table of Contents

1. Introduction ................................................................................................................ 3  
2. The iCub Head Expression Architecture .............................................................. 3  
3. Eyelid Control ............................................................................................................ 4  
4. Facial expressions electronics and control ............................................................. 5  
5. Programming Interfaces .......................................................................................... 7  
   5.1 Low-level interface .......................................................................................... 7  
      5.1.2 Operation with YARP tools ................................................................. 8  
   5.2. High-Level Interface ....................................................................................... 9  
6. Control Board ......................................................................................................... 10  
   6.1 Driver Installation .......................................................................................... 11
1. Introduction

The aim of this document is to describe the iCub Head Expression system (including also the eyelid movements), at both a user and developer perspectives.

The document is organized in the following sections:
- Overall system architecture.
- Mechanical Components and eyelid Assembly
- Electronics
- Programming Interface

2. The iCub Head Expression Architecture

The iCub head expression systems are composed by two major subsystems. The first subsystem consists in parts with a set of LEDs that allow for the display of facial features (eyebrows and mouth). The second subsystem is the eyelid mechanism activated by a servomotor. These subsystems are connected to a control board that communicates with the iCub computer via USB. The four subsystems are listed below:
- EL - The Eyelids subsystem (servo motor).
- LEB - The Left Eyebrow subsystem (led display).
- REB - The Right Eyebrow subsystem (led display).
- M - The Mouth subsystem (led display).

All subsystems are interfaced with a Control Board (based on a PIC) to a PC. Figure 1 shows the overall system architecture.

![Figure 1. Overall System Architecture, indicating the number of control wires from the control board to each subsystem.](image)
3. Eyelid Control

The eyelid mechanism is attached to the iCub face and is controlled by a servomotor, as illustrated in Figure 2.

![Figure 2: The Eyelid Mechanism](image)

The servomotor is a standard model from Futaba and controlled with a PWM input. The motor is driven by a PWM control signal with a frequency of 50Hz. The duration of the pulse determines the final position of the motor shaft according to the following table. The motor will move to the “0” position when the pulse has a width of 1ms. The position of “90º” is reached for a pulse duration of 2 ms.

<table>
<thead>
<tr>
<th>Pulse duration [ms]</th>
<th>Motor shaft position [deg]</th>
<th>Eyelids</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.9</td>
<td>90</td>
<td>Fully OPEN (100%)</td>
</tr>
<tr>
<td>1.72</td>
<td>75</td>
<td>Half-open (50%)</td>
</tr>
<tr>
<td>1.54</td>
<td>60</td>
<td>CLOSED (0%)</td>
</tr>
</tbody>
</table>

Motor: Futaba S3111; Powered at 4.8V.
## Facial expressions electronics and control

<table>
<thead>
<tr>
<th>Left eyebrow</th>
<th>Port#</th>
<th>LED’s</th>
<th>$I_{Tot.}$</th>
<th>$I_{led}$</th>
<th>Right eyebrow</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>15mA</td>
<td>3mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>15mA</td>
<td>3mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>15mA</td>
<td>3mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>15mA</td>
<td>3mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>5</td>
<td><del>GND</del></td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

---

---

---

---
<table>
<thead>
<tr>
<th>MOUTH</th>
<th>Port#</th>
<th>LED’s</th>
<th>$I_{\text{Tot.}}$ (ma)</th>
<th>$I_{\text{led}}$ (ma)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Mouth Port# 1 Diagram" /></td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td><img src="image2.png" alt="Mouth Port# 2 Diagram" /></td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td><img src="image3.png" alt="Mouth Port# 3 Diagram" /></td>
<td>3</td>
<td>3</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td><img src="image4.png" alt="Mouth Port# 4 Diagram" /></td>
<td>4</td>
<td>8</td>
<td>24</td>
<td>3</td>
</tr>
<tr>
<td><img src="image5.png" alt="Mouth Port# 5 Diagram" /></td>
<td>5</td>
<td>2</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td><img src="image6.png" alt="Mouth Port# 6 Diagram" /></td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td><img src="image7.png" alt="Mouth Port# 7 Diagram" /></td>
<td>7</td>
<td><del>GND</del></td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expression</th>
<th>Left Eyebrow</th>
<th>Right.Eyebrow</th>
<th>Mouth</th>
<th>Eyelids</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neutral</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>80%</td>
</tr>
<tr>
<td>Talking</td>
<td>*</td>
<td>*</td>
<td>{2,4,5}$&lt;-&gt;$ 3</td>
<td>80%</td>
</tr>
<tr>
<td>Happy</td>
<td>3</td>
<td>3</td>
<td>1,2,3</td>
<td>100%</td>
</tr>
<tr>
<td>Sad</td>
<td>2</td>
<td>2</td>
<td>3,5,6</td>
<td>30%</td>
</tr>
<tr>
<td>Surprised</td>
<td>1</td>
<td>1</td>
<td>2,4,5</td>
<td>100%</td>
</tr>
<tr>
<td>Evil</td>
<td>4</td>
<td>4</td>
<td>1,2,3</td>
<td>30%</td>
</tr>
<tr>
<td>Angry</td>
<td>4</td>
<td>4</td>
<td>3,5,6</td>
<td>30%</td>
</tr>
<tr>
<td>Shy</td>
<td>2</td>
<td>2</td>
<td>1,2,3</td>
<td>30%</td>
</tr>
<tr>
<td>Cunning</td>
<td>2</td>
<td>4</td>
<td>1,2,3</td>
<td>60%</td>
</tr>
</tbody>
</table>

* - Don’t care.
5. Programming Interfaces

In this section we describe how to command the several iCub Head Expressions Subsystem.

We consider two levels of command interfaces:
- A Low-Level interface, where commands are ASCII characters sent over a serial connection, and define the state of the individual sub-systems with high granularity (resolution). The interface at this level is rigid because it is hardcoded in the control boards firmware.
- A High-Level interface, where compound expressions for the whole system are sent, with low granularity. The interface at this level is very flexible because it is defined at the software level (source code and configuration files).

5.1 Low-level interface

In this section we define the low-level interface implemented in the PIC control board. At this level, all sub-systems are controlled at a very low abstraction level to provide the highest possible flexibility. Higher abstraction levels will be implemented in the High-Level interface.

The Eyelids (EL) subsystem can be controlled continuously from a LID_OPEN state to a LID_CLOSED state. Commands are given to open/close the eyelids from totally closed (0% excursion) to totally open (100% excursion), with 127 levels of discretization.

The Left/Right Eyebrow (LEB/REB) and Mouth (M) subsystems are composed by 6 independent sections capable of forming several expressions. Each section is called a PORT and each port is controlled independently by the PIC interface.

To facilitate test and debugging with tools like “Hyperterminal”, all communications with the PIC are made via ASCII characters.

Commands are composed by 1 or 3 characters.

- The first character is the address of the sub-system:
  - ‘S’ – for the eyelids (servo)
  - ‘L’ – for the left eyebrows.
  - ‘R’ – for the right eyebrows.
  - ‘M’ – for the mouth.
- The second and third characters are the hexadecimal code of a byte where each bit has the following interpretation:
  - For the eyelid sub-system:
    | BIT7 | BIT6 | BIT5 | BIT4 | BIT3 | BIT2 | BIT1 | BIT0 |
    | QUERY | VAL6 | VAL5 | VAL4 | VAL3 | VAL2 | VAL1 | VAL0 |
  - For the other subsystems:
    | BIT7 | BIT6 | BIT5 | BIT4 | BIT3 | BIT2 | BIT1 | BIT0 |
    | QUERY | NA | PORT6 | PORT5 | PORT4 | PORT3 | PORT2 | PORT1 |
QUERY – If set, determines that the corresponding subsystem state should not be changed and that the state should be communicated back to the client, as a three chars string, with the address in lower case and the state in a two chars hexadecimal code.

VAL6-0 – 7 bit integer with values between 0 and 127. If greater than 100, saturates to 100. Determines the opening percentage of the eyelids (between 0% and 100%).

NA – Not used. Reserved for future usage.

PORT6-1 – If set, and QUERY reset, turn on the corresponding led port. If not set, and QUERY reset, turn off the corresponding led port. If QUERY set, do not change the state of the port.

Examples:
SFF – read state of eyelids
M05 – turn mouth led ports 1 and 3

There are two special commands to tune the upper and lower limits of the eyelids. These commands are composed by three characters. The first character defines which limit is being set:
‘E’ – sets the servo angle corresponding to the maximum eyelid opening, in percentage of the full servo excursion.
‘U’ – sets the servo angle corresponding to the minimum eyelid opening, in percentage of the full servo excursion.

The second and third characters are the percentage value, between 00 and 99.

Examples:
E70 – sets the maximum eyelid opening to 70% of the servo excursion.
U40 – sets the minimum eyelid opening to 40% of the servo excursion.

Commands with only 1 character are used for initialization, diagnostics and termination:
- ‘I’ – Instructs the PIC to start the initialization sequence: All sub-systems are turned on for one second (all ports on and eyelids at max aperture), then all subsystems are turned off (all leds off and eyelids closed), and a string ‘ACK’ is sent back to the PC.
- ‘Z’ – All subsystems are turned off, and a string ‘ACK’ is sent back to the PC.
- ‘X’ – Cancels current command string, for example R0XL0A – an initial command was being erroneously set for the right eyebrow but was cancelled and the left eyebrow was commanded to lit ports 2 and 4.

5.1.2 Operation with YARP tools

The interface can be operated at the low-level with YARP tools. Combined use of the serial device drive network wrapper and yarp write allows sending low-level commands to the control boards. For example, if the control board is installed in
serial port COM4, the following sequence of commands will allow sending commands to the expressions system:

```bash
>> yarpdev --device serialport --comport COM4 --name /exprcontrol
>> yarp write ... /exprcontrol/in
```

Then in the yarp write console you can write the ASCII characters to send. For example, the following strings will set the SAD emotion:

```
S30
L02
R02
M64
```

It is possible to check the control board acknowledges and replies by using the yarp read utility:

```bash
>> yarp read ... /exprcontrol/out
```

### 5.2. High-Level Interface

A High-Level API is provided through a specialized YARP module called “emotionInterface”. This module opens a RPC port and accepts the following high level commands:

- **set** `<subsystem> <emotion>`
- **get** `<subsystem>`

The subsystem string can be one of the following:

- **mou** - for the mouth
- **eli**  - for the eye lids
- **leb**  - for the left eyebrow
- **reb**  - for the right eyebrow
- **all**  - for the whole system

The subsystem emotion can be one of the following:

- **neu**  - for Neutral
- **tal**  - for Talking
- **hap**  - for Happy
- **sad**  - for Sad
- **sur**  - for Surprised
- **evi**  - for Evil
- **ang**  - for Angry
- **shy**  - for Shy
- **cun**  - for Cunning
6. Control Board

The control board commands the left and right eyebrows LEDs, the mouth LEDs and the eyes servo motor. It connects with the computer using a USB PORT. The microcontroller that performs this task is a PIC18F2550 USB microcontroller from Microchip. This PIC has 32K Bytes of Flash memory, runs at 8MHz and implements an USB 1.1 slave communication.

The board is powered both by an external 12V power supply and the 5V of the USB host. The regulated 5V from the USB is used to power the microcontroller and also to generate the 3.3 regulated voltage necessary for the USB module of the PIC. The external 12V are used to power the LEDs and are regulated to 5V to power the servo motor. This 5V source is not used to power the electronics because the servo would introduce noise in the powering system.
Two 8 channel led drivers udn2982 are powering the leds. The udn2982 allows the high-side switching of the leds, separating the high power of the leds from the 5V of the logic. The PIC will control the lighting actuating directly in the 8 input pins of the udn2982 having the possibility to actuate 14 different sets of leds. In order to reduce the current consumption it will be placed a resistor between the leds and the udn2982. The PIC will generate a PWM control signal to control the servo motor, using his own PWM module generator.

The designed board has 40x50mm of dimension, as shown below.

6.1 Driver Installation

The control board connects to a host PC via the USB interface. In Linux, the device is automatically recognized as a Microchip device. In Windows, it is required to use the driver provided by Microchip (mchpcdc.inf), which is included in the archive of the board’s firmware.