Low complexity image recognition algorithm for handheld applications

P. Ayyalasomayajula, S. Grassi, and P.-A. Farine

Institute of Microengineering, Electronics and Signal Processing Laboratory, Ecole Polytechnique Fédérale de Lausanne,
EPFL STI IMT ESPLAB, Rue A.-L. Breguet 2, 2000, Neuchâtel, Switzerland.
phone: + (41) 32 718 3425, fax: + (41) 32 718 3402, email: pradyumna.ayyalasomayajula@epfl.ch
web: http://esplab.epfl.ch

***Abstract*—We propose a low complexity image recognition algorithm based on Content Based Image Retrieval (CBIR) suitable for handheld applications. The target application is an Alternative and Augmentative Communication (AAC) device used in speech rehabilitation and education. The device recognizes images (pictograms and pictures) and plays a sound message associated with the recognized image. Experimental validation of the proposed algorithm using MATLAB and its DSP implementation is presented.**

Keywords:* *Image retrieval; Content based image retrieval; CBIR; Alternative and Augmentative Communication; AAC; Handheld image recognition system.

I. Introduction

The field of electronic aids for disabled people is growing constantly and many innovations are added every year. In particular, there is an increasing need for electronic aids in Alternative and Augmentative Communication (AAC). Further, the use of picture symbols (pictograms) has proved successful in clinical studies to improve the communication for people with mild-to-severe speech-impairments. Communication boards, i.e., arrangements on a physical support of pictograms that convey graphical messages, are widely used for this purpose.

Content Based Image Retrieval (CBIR) has gained a lot of interest over the last two decades [1], [2]. The need to search and retrieve images from databases, based on information ("features") extracted from the image itself, is becoming increasing important. Most of the CBIR systems require large computations, preventing their use in handheld devices. In this paper we propose a low complexity Content Based Image Retrieval (CBIR) method for image recognition, applied to a handheld device called PictoBar (see Section II) used for therapy based on AAC. Emphasis is placed on the description of the image retrieval method, its experimental evaluation in MATLAB and finally its DSP implementation.

The paper is organized as follows. A brief description of the device is presented in Section II. The proposed algorithm for image recognition is described in Section III. Section IV presents experimental evaluation, and Section V explains DSP implementation. Conclusions and Future work are discussed in Section VII.

II. DEVICE DESCRIPTION

PictoBar [3] is a portable and rugged AAC device with image (pictures and pictograms) recognition capabilities, used as communication aid and for therapy of speech impaired people. Once an image is recognized, PictoBar plays a pre-recorded sound message, associated to the recognized image. PictoBar should have a fast response, thus the image recognition should be performed within one second. In order to be recognized, an image (picture or pictogram) must have been previously stored into the device database.

A. Device Hardware

The hardware block diagram of PictoBar is shown in Figure 1. The main components are:

- *Micro-camera*: used to acquire the images printed on communication boards. The camera is a fixed focus OV7675 from OmniVision with VGA resolution (640x480 pixels).
- *Processing unit*: is a DaVinci DM6446 dual core processor from Texas Instruments [4]. This processor has an ARM9 core and a fixed point C64x+ DSP core. The application runs on the ARM core, featuring user interface, peripheral control and overall system management. The image recognition algorithm runs on the DSP core.
- *Audio I/O Interface*: consisting of an Audio Integrated Circuit, a microphone for audio acquisition and a speaker for audio playback.
- *USB Interface*: used to connect PictoBar to an external PC, for firmware updates and for training new images into PictoBar database.
- *Memory*: PictoBar contains built-in memory (Flash NAND and DDR2-SDRAM) as well as detachable

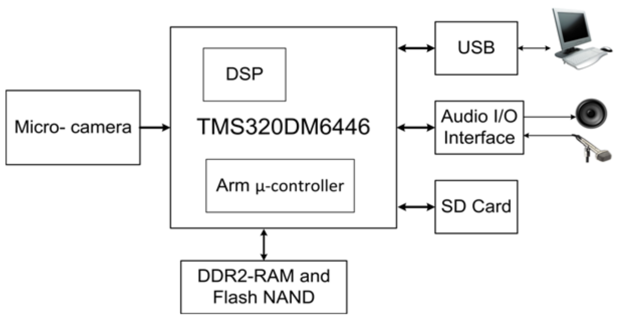

Figure 1: Hardware block diagram of PictoBar

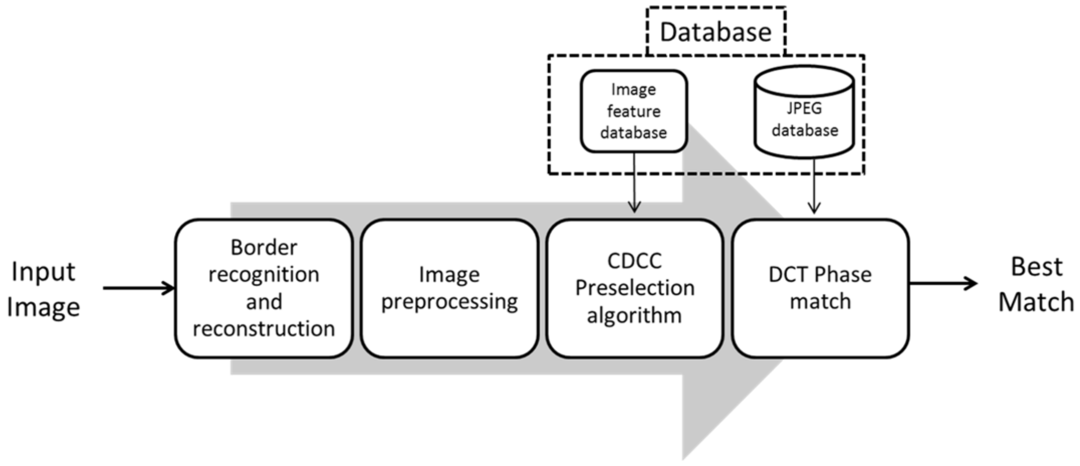

Figure 2: Algorithm overview

memory (SD card). The FLASH NAND is used to store the programs. The SD Card is used to store the database containing the JPEG compressed images and their pre-extracted features, together with the voice messages associated to each image.

III. Proposed image recognition algorithm

The proposed image recognition algorithm consists of three stages, which are shown in Figure 2 and explained as follows.

A. Border Recognition, Reconstruction and Preprocessing (BRRP)

This first stage is used to align and crop the input "query image" i.e., the image to be recognized which is acquired with the camera. The images in the communication board are, by convention, printed with a rectangular border enclosing its content (see Subsection IV.B). The enclosing border is detected using line detection, based on modified Hough transform. The query image is then aligned and cropped. Then, image pre-processing such as histogram stretching is applied. This alignment of the image is especially necessary for the third stage of the recognition algorithm (see Subsection III.C) which is not invariant to Rotation, Scale and Translation (RST).

In a case where imposing a printed rectangular border is not feasible, we can align the images using Fourier-Mellin Transform [5]. Initial development was done with this solution, but later the solution with a rectangular border was found to be more robust and should be used wherever possible.

B. Color Density Circular Crop (CDCC)

This second stage, is a pre-selection algorithm called "Color Density Circular Crop" (CDCC) [6] which is used to decrease the complexity. CDCC is a less complex but also less robust algorithm, which searches the database and preselects the 50 images which are closest to the query image.

CDCC uses as feature the color proportions within concentric circular zones of the image. To compute the CDCC features of an image we first perform edge detection using canny edge detector. Once the edge pixels in an image are found, the center of a rectangle which is formed by using the extrema edge pixels in both axes is calculated. Along with this the distance from this center to the farthest edge pixel is computed. This distance and center define a circle which tightly encompasses all the edge pixels. This circle is then divided into four concentric regions and the color density of the red, green and blue channels for each concentric region is calculated. These color densities are assembled in a feature vector of length $L = 12$ which uniquely represents the image.

For every image stored in the database, the feature vector is pre-calculated and stored. When a query image is presented, the feature vector of the query is calculated and compared with the feature vector of each of the images stored in the database. As the feature vector length is small and also low in complexity to compute, the CDCC algorithm is low in complexity. Additionally, this algorithm is robust to rotation, translation and scaling, but is sensitive to variations in lighting.

C. DCT Phase Match (DCTPM)

The third stage is the base recognition algorithm, which is referred to as "DCT phase match" (DCTPM) [7]. This algorithm is accurate and robust to lighting variation but computationally expensive and not RST invariant. To reduce the overall complexity of the algorithm while keeping good accuracy, DCT phase match is only performed on the reduced database of the 50 images preselected by the CDCC algorithm.

DCTPM searches for the best match of the query image into the database using correlation on the DCT phase of the 8x8 blocks of the images, and therefore is compatible with the JPEG compression standard.

IV. Experimental evaluation of the image recognition algorithm

A. Databases

Two databases, one for pictograms and one for pictures were used during the tests. The pictogram database contains 1500 pictograms from the Picture Communication Symbols set [8]. The pictograms were stored in JPEG format with 85% quality, and QVGA resolution (320 x 240). The picture database consist of the 1000 pictures of the COREL photograph database set used in [9] which contains JPEG compressed color images with QVGA resolution. We refer to

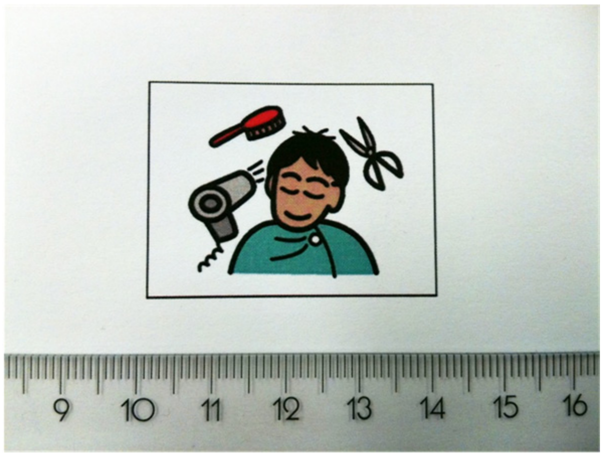

Figure 3: Example of printed pictogram with border

these two databases as the "clean pictogram database" and the "clean picture database".

B. Camera setup and real databases

To reproduce the conditions in the final device, a fixed focus OV7675 camera was fixed at a distance of 9.2 cm from the image. Under this condition, a VGA resolution acquired image corresponds to a rectangle of 6 x 8 cm. A subset of 50 representative pictograms and 50 representative pictures was selected from the 1500 pictograms and 1000 pictures of the "clean database" and were printed on paper with size of 3 x 4 cm. To provide a reference for alignment, a 3 x 4 cm rectangle enclosing the image was printed as shown in Figure 3. The 50 printed pictograms and the 50 printed pictures were then acquired with the camera, from MATLAB. These acquired images constitute respectively the "real condition pictogram database" and the "real condition picture database". Figure 4 shows the subset of pictograms and pictures used to produce the real condition database.

In preliminary tests, we found that the image recognition algorithm did not perform well with pictures. This was due to problems with color space matching under different illumination conditions for the CDCC pre-selection algorithm. To solve this problem a LED crown with a diffuser was mounted on the camera to provide uniform lighting. The camera setup with LED crown is shown in Figure 5.

C. Image recognition algorithm validation

The proposed image recognition algorithm was implemented in MATLAB. For testing, we have used the two clean databases presented in Subsection IV.A. Every image within the database is successively used as query, running the search for this image on the entire database. Tests were also performed with different rotations of the query image at 30, 60 and 90 degrees. The recognition accuracy is calculated as the proportion of cases in which the best match corresponds to the query image, with respect to the number of trials. The results for these tests showed 100% accuracy, hence validating the image recognition algorithm. We also performed tests with the real condition databases (see Subsection IV.B) and no RST. These tests also resulted in query image found as best match in every case from the test database. We can see that the proposed image recognition algorithm performs very well with both clean databases and real databases.

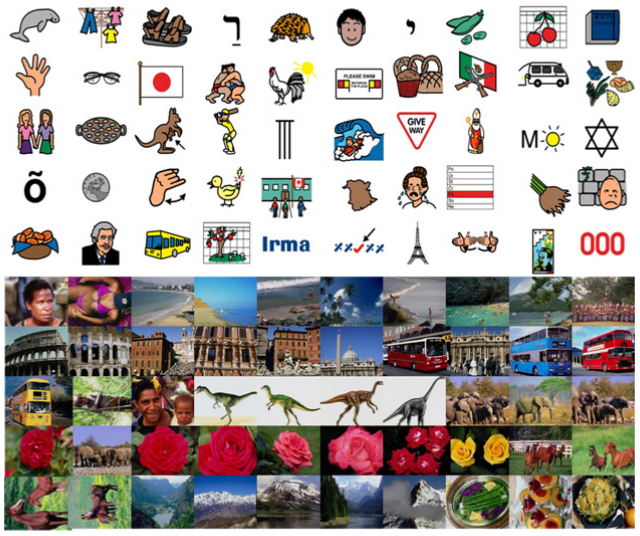

Figure 4: The subset of 50 pictograms and 50 pictures used for producing the real databases

V. DSP IMPLEMENTATION

After validating the algorithm in MATLAB as explained in Section IV.C, the algorithm was ported on a TMS320DM6446 DaVinci processor [3]. We used the Codec Engine framework [10] from Texas Instruments (TI) for this DSP implementation. The MATLAB implementation is used as a reference throughout the implementation.

We have implemented the different blocks of the algorithm in C language using fixed point arithmetic. This was partly done under Code Composer Studio (CCS v4) along with Blackhawk USB-JTAG emulator as debugging tool. To implement the BRRP, CDCC and DCTPM blocks, we used as much as possible the functions from the Image library (IMGLIB2) and the Vision library (VLIB) provided by Texas Instruments [11], [12], such as Hough transform, Canny edge detection, 8x8 block DCT calculation and image quantize for JPEG quantization.

The algorithm blocks to be implemented in the Codec Engine framework are BRRP, CDCC, DCTPM and JPEG decoder. Each of the blocks was implemented as a separate Codec Engine "codec". The BRRP, CDCC and DCTPM codecs were implemented using the IUNIVERSAL xDM standard interface [10]. The JPEG decoder [4] which is provided by Texas Instruments is delivered as IIMGDEC1 xDM codec. A single Codec Engine server configured to include these four codecs is made available to the application. We have used the Codec Engine GenCodecPkg wizard [13] to generate each codec package containing starter code and then we have added the C code of the algorithm blocks into it. The testing application for each codec using File I/O was written by modifying the universal copy example delivered with the Codec Engine software.

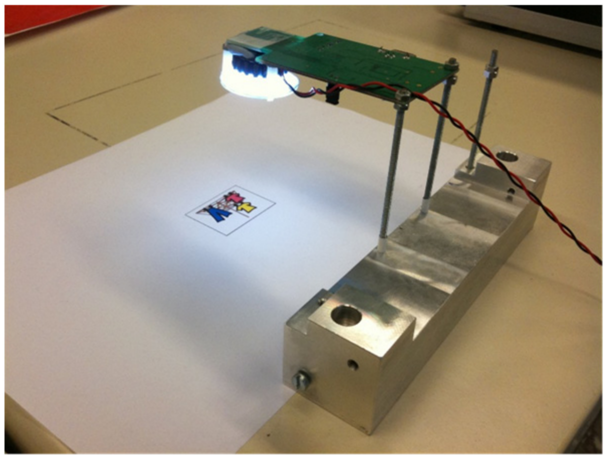

Figure 5: Camera with LED crown

A. Testing

We have put the file system of the DM6446 board on a server with a fixed IP address and mounted it using NFS. The board, configured to boot from Flash NAND, was also connected to the network and was assigned a fixed IP address. This allowed the access from remote location to the file system via Samba or NFS and facilitated remote login and execution on the board using SSH or Telnet.

This setup allows sharing of the board between users and the possibility to perform extensive testing automatically from MATLAB: first the input files are written from MATLAB on the board file system, execution of the application on the board is controlled from MATLAB using SSH, and the output files are read by MATLAB from the board file system to analyze the results. Using this development setup, we have extensively tested the codecs with images from the "clean database" (see section IV.A. By comparing the results from the DSP implementation with the results from the MATLAB implementation, we have verified the proper functioning of the DSP implementation, and that the losses due to the fixed-point arithmetic of the C64x+ do not affect the performance.

The results for CDCC and DCTPM blocks tested on the DSP with "clean database" showed the same performance as MATLAB tests i.e., 100% accuracy. The recognition time was well within the specified limit of one second.

VI. Ongoing and future work

Ongoing and Future work include:

- Testing new camera optics with a higher viewing angle to reduce the camera height, for ergonomic reasons.
- Exploring ways to make CDCC insensitive to lighting variation, e.g. by using color spaces with no intensity information. This would improve robustness and avoid frequent calibration of the device.
- Improve the preprocessing stage to correct for geometrical distortions such as barrel and tilt.
- Extensive testing of the DSP implementation with different real acquisition conditions of lighting, rotation, etc.
- Perform characterization of the resources and timing of the DSP implementation.
- Architectural optimization of the DSP implementation such as using of cache and switching off the DSP when not used, to further improve the response time and reduce the complexity.

VII. Conclusions

A low complexity image recognition system for application in a handheld device for Alternative and Augmentative Communication was presented. The device recognizes images and plays a sound message associated with the recognized image. Experimental validation of the image recognition system in MATLAB was presented. DSP implementation using Codec Engine framework was also presented along with a procedure to systematically test the DSP implementation.

Acknowledgment

This work was partly supported by the Swiss Federal Office for Professional Education and Technology (OPET) through the Innovation Promotion Agency (CTI) under the Grant CTI 8811.2 PFNM-NM ("PictoBar" project).

References

[1] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, "Content-based image retrieval at the end of the early years", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.22, no.12, pp. 1349-1380, Dec. 2000.

[2] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age", ACM Computing Surveys, vol. 40, no. 2, article 5, 2008.

[3] P. Ayyalasomayajula, S. Grassi et al., "Implementation of an image recognition algorithm on the DM6446 Da-Vinci Processor", Proceedings of the 4th European DSP in Education and Research Conference, pages 175-179, Dec. 2010.

[4] TMS320DM6446 Digital Media System-on-Chip - http://focus.ti.com/docs/prod/folders/print/tms320dm6446.html.

[5] P. Ayyalasomayajula, S. Grassi et al., "Low complexity RST invariant image recognition using Fourier-Mellin Transform", Submitted to EUSIPCO 2011.

[6] P. Ayyalasomayajula, S. Grassi et al., "Rotation, scale and translation invariant image retrieval method based on circular segmentation and color density", Submitted to 7th Int. Symp. on Image and Signal Processing and Analysis (ISISPA 2011).

[7] J. Bracamonte, M. Ansorge, F. Pellandini, P.-A. Farine, "Efficient compressed domain target image search and retrieval", Proceedings of the fourth International Conference on Image and Video Retrieval, CIVR'05, Singapore, pp. 154-163, July 2005.

[8] http://www.mayer-johnson.com

[9] J.Z. Wang, Li Jia, G. Wiederhold, "SIMPLIcity: seman-tics-sensitive integrated matching for picture libraries", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.23, no.9, pp.947-963, Sep. 2001.

[10] TI Codec Engine Application Developer Users Guide, Texas Instruments, edition d, September 2007.

[11] TI TMS320C64x+ DSP Image/Video Processing Library (v2.0.1), Texas Instruments, edition a, May 2008.

[12] TI Vision Library (VLIB) Application Programming Interface Reference Guide, Texas Instruments, edition c, November 2009.

[13] http://processors.wiki.ti.com/index.php/Codec_Engine_GenCodecPkg Wizard FAQ