# Design and Implementation of a Software Presenting Information in DVB Subtitles in Various Forms

K. Sinan Yildirim, Aybars Ugur, *Member*, IEEE, A. Cumhur Kinaci

**Abstract** — *Subtitle data carried by MPEG-2 Transport Stream includes texts of dialogs in bitmap graphics format. However, this data is not reachable for the blind audience. In addition, carrying extra compatible data with subtitle packets can be helpful for testing subtitle software automatically and for presenting live content (i.e. latest news or special advertisements). In this paper, we propose a method for converting subtitle data into text format by recognizing characters using neural networks. The converted data can be used for blind television users and testing subtitle software automatically. We also define a new subtitle segment for carrying test comparison data and a new subtitle page for representing live content. We present our implementation on Linux platform.*[1]

**Index Terms — DVB Subtitles, live content, automatic testing, neural networks, character recognition.**

## I. INTRODUCTION

*Digital Video Broadcasting* (DVB) is a suite of internationally accepted open standards for digital television. This technology relies on the viewer having a suitable digital receiver which is able to visualize many ancillary services including subtitles [1].

DVB subtitle data are carried through MPEG-2 *packetized elementary streams* (PES) that carry page composition, region composition, CLUT definition and object data segments. The data carried in these segments include rectangular regions that will be presented on a subtitle page, run-length coded bitmapped graphics that will be placed in that regions and color lookup tables associated with the bitmapped graphics. The PES stream is decoded by a subtitle engine on digital receiver side and the produced subtitle images are rendered into the on screen display (OSD) [2].

DVB subtitles provide access to television for the hard-of-hearing and for the deaf people who can read. They are also useful for accessing multilingual translations of dialogs. However, blind people may also need a way to get multilingual translation of the dialogs as well as other users may want to get subtitle data of a program, i.e. news, in a text format for a specific purpose.

In this paper, we propose a method for converting DVB subtitles from bitmapped graphics to text format. With using the converted data, we can translate subtitle texts to Braille alphabet. We also define a new subtitle segment to carry test data and a new subtitle page to carry live content data. By carrying test data through this new segment, we propose an automated testing method that can be used to test subtitle engine software independently. By carrying extra data through live content pages, we decode and present live content information (i.e. latest news, special advertisements) which is useful for most of the television users. Fig.1. gives an overview of using dvb subtitles for specific purposes we will describe.

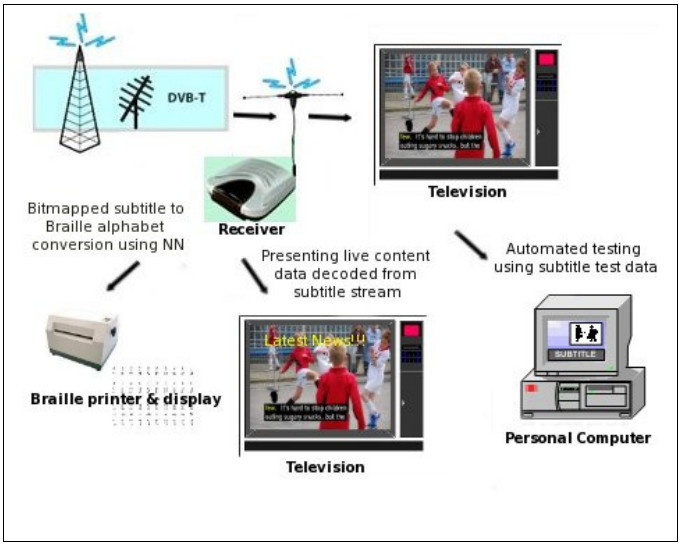


**Fig. 1. Different usage of subtitle data carried through PES packets**

This paper is organized as follows. Section II describes steps for converting subtitle data to text format. Method for testing subtitle engine in an automated way is described in Section III. Section IV describes carrying live content data with PES packets. We also implemented a DVB subtitle engine that has capability to convert subtitle data from bitmap to text on a Linux platform which is shown in Section V. Section VI is the conclusion.

## II. BITMAP TO TEXT CONVERSION

Many character and number plate recognition studies based on neural networks are proposed in the literature [4], [5], [6] and [7]. In order to convert subtitle bitmaps into the character strings, the subtitle engine we implemented can return us bitmap data of each region in a page separately. After we get

[1] K.S. Yildirim is with the Computer Engineering Department, Ege University, Bornova, Izmir, 35100 TR (e-mail: sinan.yildirim@mail.ege.edu.tr).

A. Ugur is with the Computer Engineering Department, Ege University, Bornova, Izmir, 35100 TR (e-mail: aybars.ugur@ege.edu.tr).

A.C. Kinaci is with the Computer Engineering Department, Ege University, Bornova, Izmir, 35100 TR (e-mail: cumhur.kinaci@ege.edu.tr).

Contributed Paper
Manuscript received September 25, 2007

data of the bitmap inside each region, we follow the conversion steps which are described below.

*A. Conversion Steps*

The conversion procedure shown in Fig.2.consists of four different steps.

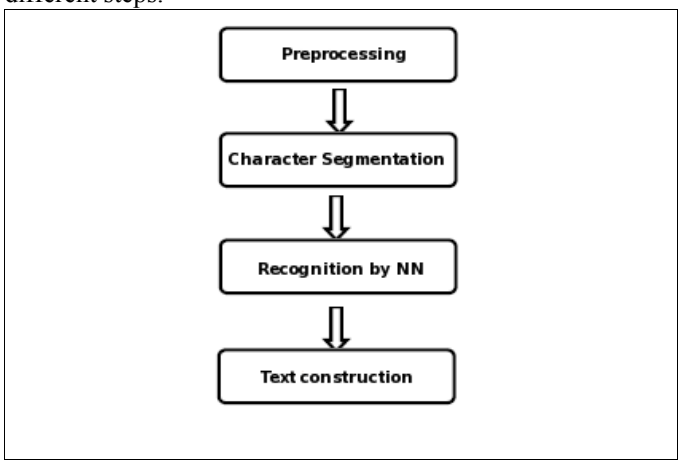


**Fig. 2. Steps for recognizing subtitle characters**

*Step 1: Subtitle bitmap preprocessing.* The bitmap data of each region in a subtitle page includes index values to the color lookup tables which are also carried by PES stream. Before starting character segmentation, we convert each value in the bitmap data to the black and white color values. This preprocessing makes character recognition in the following steps easier.

*Step 2: Character segmentation.* After subtitle region is converted to a bitmap that includes only black and white color values, characters inside the bitmap are collected by computing widths and heights of the characters using smearing algorithm.

Step 3: *Character recognition using neural network.* The collected characters are then fed to the neural network which has previously been trained. Neural network forwards the recognized characters to the following processing layer.

*Step 4: Text construction.* The recognized characters are combined to form the character string of the region bitmap. The character strings are buffered inside the subtitle engine.

*B. Neural Network Structure*

A neural network is a massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making available for use [3]. In this work, neural networks technique is used in for character recognition process in our embedded subtitle engine software.

For the character recognition problem, a multilayer perceptron with one hidden layer is used which is shown in Fig.3. Characters in our sample input data is a 20x20 pixels bitmap, so that the input layer should have 400 neurons. Hidden layer has 30 and output layer has 26 neurons which are equal to the number of classified letters and backpropagation is chosen for the learning algorithm.

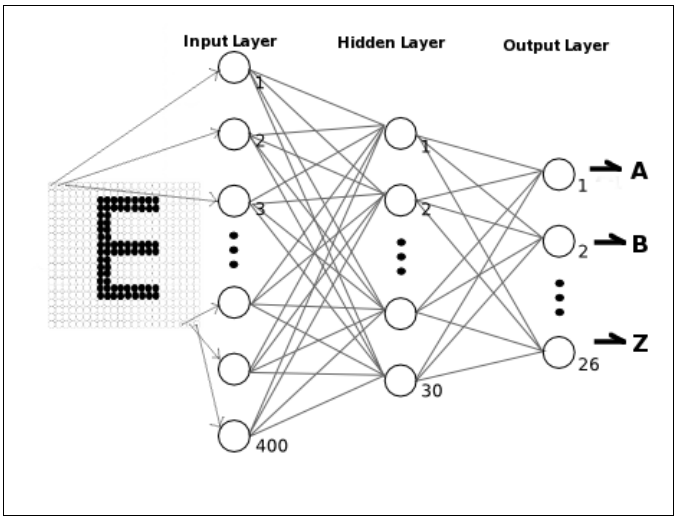


**Fig. 3. Structure of neural network embedded in subtitle engine**

## III. USING CONVERTED DATA

As a result of embedding character recognition process inside the subtitle engine, it is more efficient and easy to detect characters inside a subtitle region. After bitmap to character string conversion, the subtitle text can be transmitted with using the serial port of the digital receiver. By implementing a simple data transmission protocol, any peripheral connected to the digital receiver can receive the text data. For example, we can connect our PC to the digital receiver via serial port, receive subtitle text of a news program and print it by using a Braille printer. This is useful for disabled people that that cannot access the broadcasted subtitle.

In addition, the converted data can also be used for specific purposes. In this section, we describe a method for testing subtitle engine software automatically using the converted data.

*A. Independent Testing Environment*

In typical digital television software, the whole system is split into tasks that run on *central processing unit* (CPU) in parallel [8]. For subtitle processing, *transport stream engine* task is responsible for collecting PES packets that carry subtitle data. *Subtitle engine* task is responsible for decoding the collected PES packets and *graphics engine* task is responsible for plotting subtitle bitmaps on display. A *system controller* task is responsible for keeping the states of the system. The interaction between system tasks is shown in Fig.4.

To test subtitle engine task independently, we must break the dependencies of our engine on the other system tasks. In order to achieve this, we developed a tool that can collect PES packets that carry subtitle data from a previously recorded stream, so that we eliminated the need for a transport stream engine task. In addition, data plotting requests of our subtitle engine are delegated to our fake graphics engine layer and we also eliminated the need for a real graphics engine. At this point, we have subtitle engine that can completely run standalone on a PC platform without having any dependency.

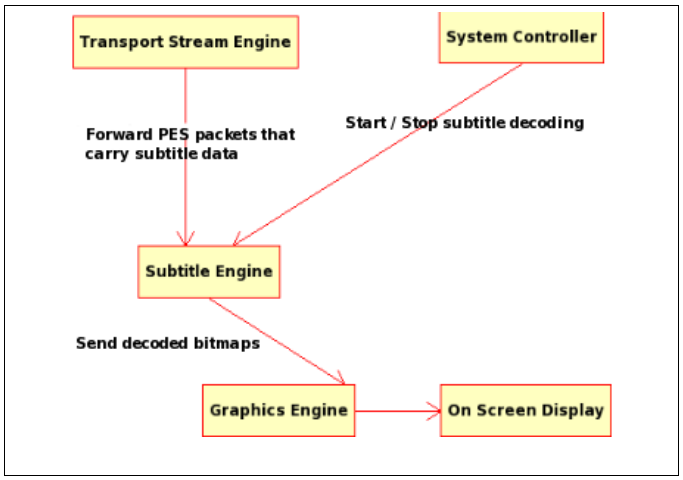


**Fig. 4. System tasks responsible for subtitle processing**

### *B. Testing Subtitle Engine Automatically*

We collected several subtitle content and produced a PES test stream in order to test our subtitle engine. To validate each PES packet inside this PES stream during the test, we previously saved the attributes of subtitle page, regions inside the subtitle page and corresponding text values of subtitle bitmaps inside the regions of each PES packet in xml format to a comparison file. After each PES packet is decoded, the comparison data of the page is read from the comparison xml file and checked. If attributes of the decoded packet is the same as saved attributes in the xml file, the test case is passed. Testing steps are shown in Fig. 5.

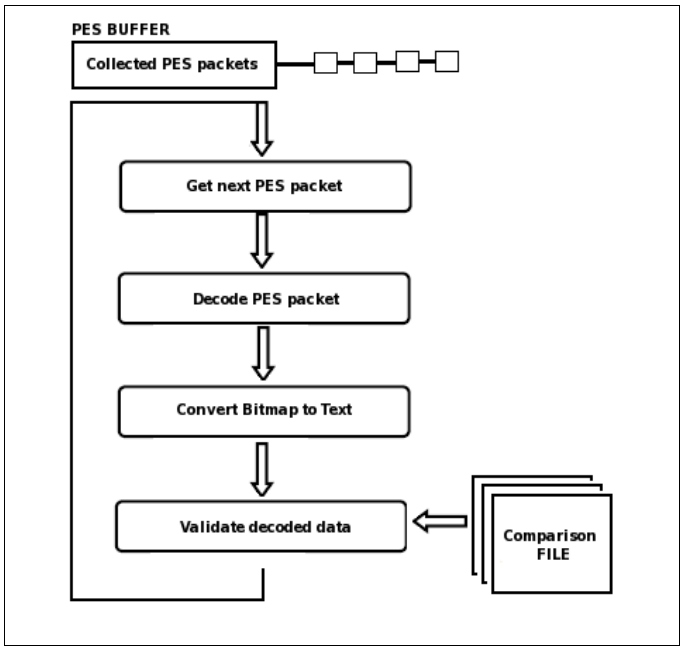


**Fig.5. Testing procedure for subtitle engine**

These steps can automatically be executed on a *personal computer* (PC) platform as well as on real hardware. The comparison data can also be embedded with the application code and the comparison operations can be done using data in memory instead of using a xml file. After each change on subtitle engine software, these automatic tests can be run to find if any change in software has affected previous successfully passed test cases.

### *C. Eliminating The Use Of Comparison File*

To prevent using previously validated comparison data file, broadcasters may also add an extra segment to each PES packet that carries comparison data for that subtitle page. The receivers can use this extra segment to validate their decoded data.

We call this extra segment *Test Comparison Segment* (TCS) and its contents are shown in Table I. We assigned 0x14 value as segment type. TCS includes page data, its region positions and texts inside the regions. We can compare our converted text data with the value broadcasted in this segment. TCS may also contain *cyclic redundancy check* (CRC) value for the current subtitle page. The receiver can compute the CRC value of its subtitle page buffer and compare it with the CRC value received via TCS. TCS contents showed in Table I also includes CRC field.

Also, the comparison text value which is carried by this segment can directly be sent to the serial port. This may eliminate bitmap to character conversion using a neural network when the data taken from serial port will be used for printing, recording, etc.

**TABLE I**
**TEST COMPARISON SEGMENT**

| Syntax | No. of bits |
|---|---|
| Test_comparison_segment() { | |
| sync_byte | 8 |
| segment_type | 8 |
| page_id | 16 |
| segment_length | 16 |
| page_time_out | 8 |
| page_version_number | 4 |
| page_state | 2 |
| reserved | 2 |
| page_CRC | 32 |
| while (processed_length < segment_length) { | |
| region_id | 8 |
| reserved | 8 |
| region_horizontal_address | 16 |
| region_vertical_address | 16 |
| region_width | 16 |
| region_height | 16 |
| region_text_data_length | 8 |
| while(processed_length<region_text_data_length) | |
| region_text_data_block() | |
| } | |
| } | |

## IV. CARRYING LIVE CONTENT WITH SUBTITLE DATA

In [8], a new MPEG-2 TS compatible table is defined to carry live content data. That table can be filtered and processed in order to present live content to the user during playback.

It is also possible to carry live content data via subtitle PES packets. While broadcasters can broadcast *TCS* that makes

testing process for digital receivers easy, we can define another page type which is compatible with ancillary and composition pages, which carries special data including latest news, special advertisements, etc. We call this page *Live Content Page*(LCP).

PID, language and page-id for the for the television service of interest are signaled in the *Program Map Table* (PMT). We can add page-id of LCP into the PMT and insert LCP data into the PES stream that carries subtitle data of that service. This is shown in Fig.6.

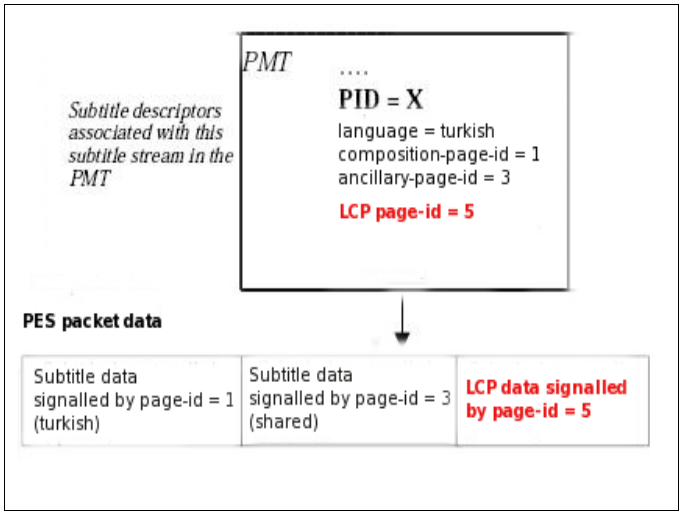


**Fig.6. LCP broadcasted in PMT of the service**

With using LCP, users can be notified instantly. Enabling and disabling LCP notifications can be done via the remote receiver and adding a simple digital menu. Since LCP is compatible with ancillary and composition pages, no extra implementation is needed on subtitle engine side.

## V. IMPLEMENTATION ON LINUX PLATFORM

We implemented a subtitle engine and a PC tool called *PES Collector* that collects PES packets and merges them into a subtitle stream on *Slackware Linux* PC platform. We also ported our subtitle engine to *MIPS* architecture. In addition, we developed a PC tool called *PES Constructor* for generating PES packets that carry compose TCS.

For character recognition process, we used Fast Artificial *Neural Network Library* (FANN). FANN is a free open source neural network library, which implements multilayer artificial neural networks in C with support for both fully connected and sparsely connected networks [9].

The code segment of the program that is responsible for creating, learning and saving the neural network with using FANN is shown in Fig. 7.

```
const unsigned int num_input = 400;
const unsigned int num_output = 26;
const unsigned int num_layers = 3;
const unsigned int num_neurons_hidden = 30;
const float desired_error = (const float) 0.001;
const unsigned int max_epochs = 500000;
const unsigned int epochs_between_reports = 1000;

struct fann *ann = fann_create_standard(num_layers,
num_input, num_neurons_hidden, num_output);

fann_set_activation_function_hidden(ann,
FANN_SIGMOID_SYMMETRIC);
fann_set_activation_function_output(ann,
FANN_SIGMOID_SYMMETRIC);

fann_train_on_file(ann, "letters.data", max_epochs,
epochs_between_reports, desired_error);

fann save(ann, "nn.net");
```

**Fig.7. Using FANN library in embedded character recognition**

In Fig.8, subtitle engine running on Slackware Linux platform is demonstrated. Subtitle stream that carries PES packets generated by *PES Constructor* and the test comparison file is given as an input. Tests can automatically run and subtitle engine provide visual input to the user.

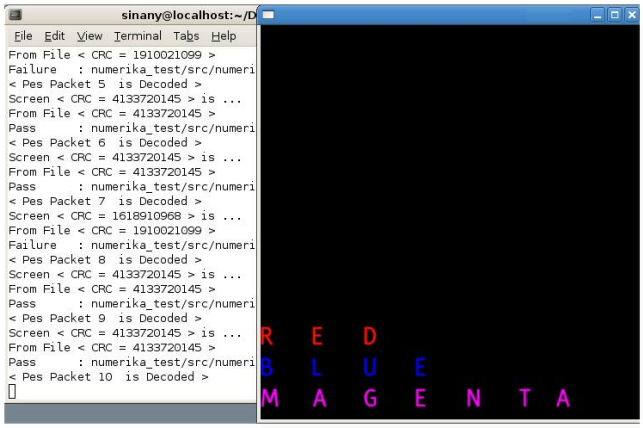


**Fig.8. Subtitle engine running on Slackware Linux**

We also generated a sample LCP page and embed its data into the subtitle stream. Our engine can successfully decode LCP content and present it without adding any extra implementation. As we stated before, adding a LCP entry to the PMT is enough for decoding LCP data of the service of interest.

## VI. CONCLUSION

In this paper, we proposed a method for converting subtitle bitmaps to text format. We showed that the converted data can be used for blind people to access subtitles of programs such as news. We described a method for automatic testing of subtitle engine software using the converted data. To prevent using previously recorded test comparison data, we also described a new subtitling segment *TCS*. We defined another

compatible page *LCP* for carrying data about the latest news and events with subtitle PES packets. We also presented sample outputs of our implementation on Linux platform.

We conclude that, embedding character recognition process inside the subtitle engine, it is easy to get and use converted data. In addition, if broadcasters start to broadcast *TCS* segments, it will be easy for digital receivers to check their decoding errors. *LCP* can also be useful for broadcasters to broadcast latest news and advertisements easily.

## ACKNOWLEDGMENT

Special thanks to Turkish Scientific and Technical Research Council (TÜBİTAK) for granting a scholarship to one of authors, K. Sinan Yıldırım.

## REFERENCES

[1] U. Reimers, "The DVB project-digital television for Europe", *DVB (Digital Video Broadcasting): The Future for Television Broadcasting?), IEE Colloquium on* (Digest No.1995/142), Page(s):1/1 - 1/7,27 Jun 1995

[2] *ETS 300 743, Digital Video Broadcasting (DVB);*Subtitle Systems, EuropeanTelecommunications Standards Institute, September 1997.

[3] Haykin, Simon,*Neural Networks: A Comprehensive Foundation (2nd Edition)*, Prentice-Hall, 842p, 1999

[4] M. Cinsdikici, **A. Ugur**, T. Tunali, "Automatic number plate information extraction and recognition for intelligent transportation system", *The Imaging Science Journal*, ISSN 1368-2199, Vol. 55, Number 2, June 2007, pp 102-113

[5] J.L.R. Fletcher, R. Kasturi, "A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images', *IEEE trans. On Pattern Analysis and Machine Intelligence*, 1998, 10/6, 910–917.

[6] H. Hontani, T. Koga, "Character Extraction Method without Prior Knowledge on Size and Position Information", *IEEE International Vehicle Electronics Conference:IVEC2001*, Tottori, Japan, September 2001, Tottori Unv, 67–72.

[7] I.P. Morns, S.S. Dlay, "The DSFPN: A new neural network and circuit simulation for optical character recognition ", *IEEE Transactions on Signal Processing* 51 (12): 3198-3209 DEC 2003

[8] E. Dogan; O.K. Erol, "Method for Providing Live Content during Playback of Recorded Streams in Per sonal Video Recorders", IEEE Transactions On Consumer Electronics, vol. 52 , issue 2, no.pp. 1253-1255, Nov. 2006

[9] http://leenissen.dk/fann/

**Kasım Sinan YILDIRIM** received the B.S. and M.S. degrees in the Computer Engineering from Ege University, İzmir, Turkey, in 2003 and 2006 respectively and continues his Ph.D. at the same department. His research interests are embedded systems, real-time systems and distributed systems.

**Aybars UĞUR** received the B.S., M.S. and Ph.D. degrees in the Computer Engineering from Ege University, İzmir, Turkey, in 1993, 1996 and 2001, respectively. He has been an Assistant Professor with the Department of Computer Engineering, Ege University, since 2001. His research interests are artificial intelligence, intelligent systems, computer graphics, algorithms and image processing. Dr. Ugur is a member of IEEE.

**Ahmet Cumhur KINACI** graduated from Department of Statistics at Middle East Technical University, Ankara, Turkey, in 2003. He received his MSc in Department of Computer Engineering at Ege University, İzmir, Turkey, in 2006. And he is currently working towards his PhD in the field of neural networks in the same department.