Modelling of Variable Message Signs for Driving Simulation

I. Pareja
Institute of Traffic and Road Safety (INTRAS)
University of Valencia
C/ Hugo de Moncada 4, 46010, Valencia, Spain
ipareja@uv.es

***Abstract*— Variable Message Signs (VMS) are dynamic items for traffic signalling and information which started to be widely introduced in the last years in the road infrastructures of the Spanish territory and other European countries. Their main purpose is to control traffic and adjust it to changes or incidents.**

They are basically used to inform drivers about any event or incident interesting for road safety or traffic fluency like traffic density, nearby car accidents, closed roads or streets, adverse atmospheric conditions, etc. Furthermore, they are frequently used to remind drivers of important traffic rules in certain circumstances.

The current paper expounds the development of an architecture to define and model the 3-D texts and graphs that are visualized on the VMS. The generated models have been included in a driving simulator that was developed by the Institute of Traffic and Road Safety of the University of Valencia after several years of collaboration among professionals from different areas.

The use of simulation gives outstanding advantages compared with on-road experimentation ones. Those advantages may be summed up in the possibility to analyse the VMS features and their influence on driving in a quicker and more economic way.

***Index Terms*— Real-time Driving Simulation, Road Safety, Virtual Reality, VMS.**

I. Introduction

It has been some time since driving simulation has become an extremely useful tool to experiment with the purpose of analysing and evaluating different traffic and road safety characteristics. In this sense, it offers a great help in many of the most relevant road safety areas like infrastructure analysis, driver evaluation, training, etc.

Among the numerous existing signalling items, Variable Message Signs represent a type of signal relatively new that has been introduced lately in the Spanish territory and other European countries. They belong to the so-called ITS (Intelligent Transport Systems) which include all the computerised technology used to control traffic and try to improve road safety.

VMS-panels are a special kind of vertical signalling that makes possible to show dynamic messages, i.e., messages that can be displayed, deleted or changed at any time (Fig. 1). Their main purpose is to inform drivers about a wide range of traffic incidents or events: traffic density, surrounding car accidents, closed roads or streets, adverse atmospheric conditions, etc. Moreover, they are usually utilized to send some advice and reminders to drivers about road rules in certain traffic circumstances.

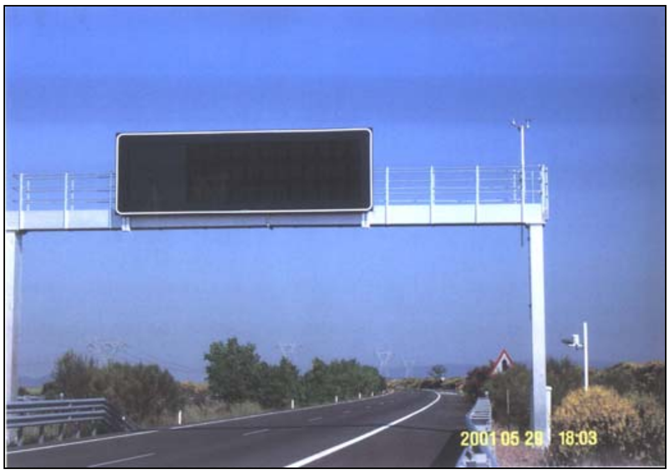


Fig. 1(a) – VMS-panel

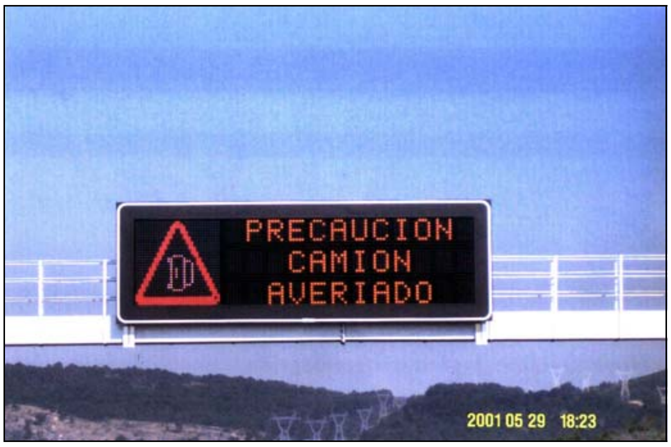


Fig. 1(b) – VMS-panel containing data

The current paper is basically focused in the development of an architecture that makes possible to integrate 3 dimensions models representing messages that can be displayed on a VMS-panel into a driving simulator. These 3D models must necessarily be dynamic, i.e., they must be able to be defined, modified, displayed or deleted at any time of the simulation application execution.

The motivation to incorporate VMS into the simulator is not only reduced to the fact that they are items of the traffic environment; there is also a great interest in analysing and evaluating their different aspects (legibility, understanding, efficiency, visibility, etc).

On the other hand, the performance of VMS analysis and evaluation in the driving simulator offers more possibilities than on-road experimentation, in fact, these kinds of experimentation can turn out to be extremely expensive or even unsustainable in certain cases (analysis of new kinds of messages, different atmospheric or visibility conditions, different locations, etc).

II. VMS FEATURES

Variable Message Signs are composed of high brightness pixel matrices. In the VMS context, an item consisted of a group of light emitting diodes (LEDs) – jointly lighted - is called pixel.

VMS pixels are grouped into matrices of different kinds and sizes. Specifically, there exist two different kinds of matrices:

- Alphanumeric matrices. These matrices can display all sorts of alphanumeric characters like letters, numbers, punctuation marks or symbols on the VMS-panel. They are usually composed of 35 pixels modules distributed in 7 rows and 5 columns (Fig. 2). These pixels can only generate light in amber.
- Graphic matrices. They can display graphs, called pictograms or icons. These pictograms correspond to traffic signs or other signs beyond the code but following a similar philosophy. They commonly consist of 1024 pixels distributed in 32 rows and 32 columns as you can see in the left side of the Fig. 2. Unlike alphanumeric matrices, their LEDs can generate amber, white, red, green and blue colours. This chromatic range makes possible to represent a great variety of graphic elements.

Attending to the quantity, size and configuration of the matrices, there currently exist different kinds of PMV-panels. Nevertheless, only one among all of them has been initially chosen to be modelled in 3 dimensions and included in the simulator. The selected panel has a configuration formed by one graphic matrix and 36 alphanumeric matrices grouped into 3 lines of 12 matrices (Fig. 2). This VMS-panel is quite usual in the Spanish and European roads, although there might be other types of panels that include more o less graphic or alphanumeric matrices.

It is important to note that the architecture to define messages which is being expounded in the present paper is valid for the type of panel modelled as well as for any other, because it mainly consists in the definition of the language elements that are used in messages (letters, numbers, images, etc.) and it is independent from the distribution and quantity of alphanumeric and graphic matrices on the panel.

III. DRIVING SIMULATOR

The driving simulator where the VMS 3D model has been integrated was developed by the Institute of Traffic and Road Safety (INTRAS) of the University of Valencia after years of collaboration among researchers of several areas like Computer Science, Psychology or Physics. This simulator is a high-performance system composed of the following parts [4]:

- A widely sensored real vehicle that makes possible normal driving by means of its controls and a communication card that receives the information from each sensor.
- A projection system consisted of a curve screen or three flat screens that cover more than 120 degrees of field of view and three high quality projectors with 1000 lumens (Fig. 3). Furthermore, the projection includes the visualisation of the three vehicle's rear-view mirrors.
- A graphic workstation Silicon Graphics ONYX2 able to generate all the images needed to represent the driving simulation scenes in real-time.
- A sound system managed by a PC running under Windows and composed of four loudspeakers and one subwoofer.

At the beginning, the implementation of the simulator software was done in C++ language and using the OpenGL Performer graphic library from Silicon Graphics. OpenGL Performer is a powerful tool to create real-time 3D visual applications. This library includes several kinds of classes to create and manage 3D text objects, some of them were used to build and represent the messages and graphs that can be

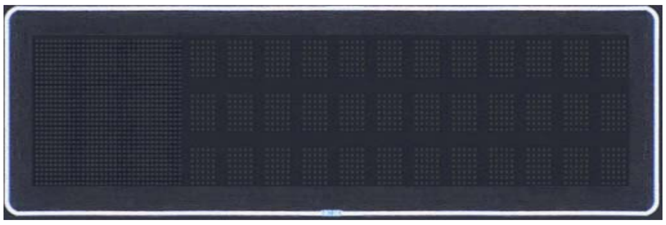


Fig. 2(a) – Simulated VMS with pixels off

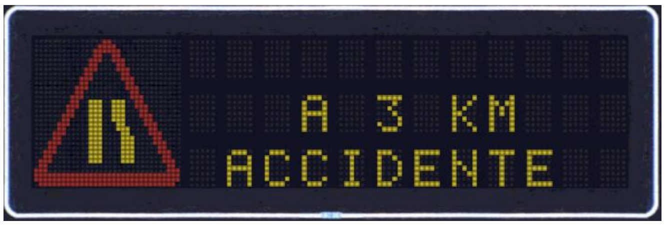


Fig. 2(b) – Example of simulated VMS showing a message

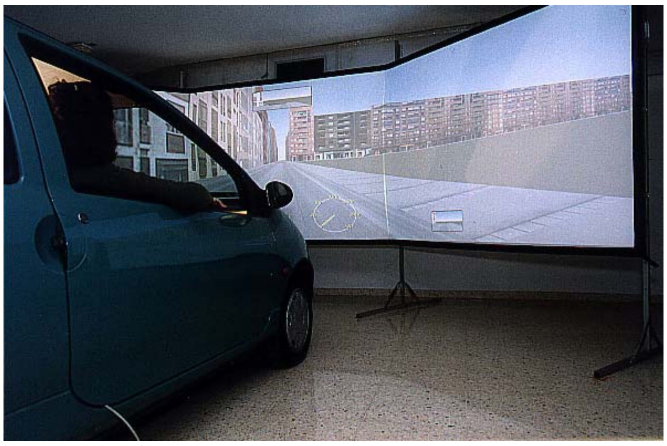


Fig. 3 – Driving Simulator of INTRAS

displayed in the Variable Message Signs panels in the simulator (Fig. 4). It used to be one of the more used libraries in the high quality visual simulation field but nowadays there are other powerful alternatives.

In this sense, this software has recently been ported to OSG (Open Scene Graph). OSG is an open source high performance 3D graphics toolkit entirely written in Standard C++ and OpenGL that runs on almost every platform. As OSG does not have any class to represent 3D text object, it has been necessary to implement these classes.

Initially, this driving simulation system was developed as an infrastructure assessment tool. In this field, driving simulation has a great projection because it can highly help to study the existing routes to be analysed or to study improvements. Moreover, it can even be used to evaluate new routes [1,2].

Subsequently, the simulator was upgraded to be used as a driver evaluation tool. The development of this evaluation system was carried out jointly with the MAPFRE Institute for Road Safety in the so-called EVICA project (Computer System for Driver Interactive Evaluation of the MAPFRE-MARES insurance company) [4].

In the EVICA system, driver evaluation is performed thanks to the creation of driving scenarios where individuals must drive along. A large quantity of measurements is taken in each scenario situation and all way long. These measurements are related to paths, times, distances, actions and decisions and make possible to perform, using an expert system for analysis, a driving style evaluation by means of an automatically printed report. This evaluation can offer drivers some information and advice to help them improve certain aspects where deficiencies have been detected [3] and [4].

IV. 3D TEXT FONT

The definition of the 3D models of the characters that compose the VMS messages has been performed through the creation of a 3 dimensional text font. The so-called OpenGL Performer class pfFont has been used to create this font. This class includes functions and properties in order to generate user-defined 3D text. Afterwards, this text can be visualised

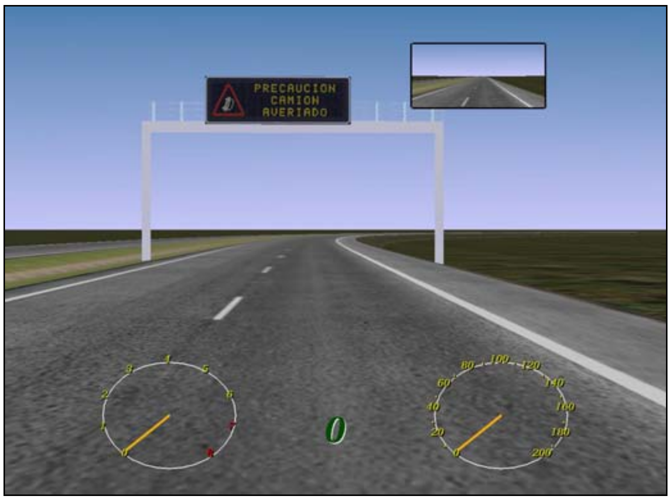


Fig. 4 – Driving Simulator with VMS-panels snapshot

in real-time.

An instance (object) of the pfFont class is basically a collection of 3D objects or groups of objects that represents a certain character. In addition to the character typical definition, this class can include other metric information like the one related to the gap or distance between characters for example [5].

It is also important to point out that the pfFont class is a passive class in the sense that it can not be drawn, because it simply provides a set of 3D characters to a high level class called pfString. This class can generate strings of 3D characters using a 3D font previously defined. An object of the pfString class represents an array of pointers to elements of the 3D font [5]. On the last level there is the pfText class, this class can join several pfString to compose a 3D text which can be visualised.

As we said before, the pfFont class has been used to define the 3D font that is needed to represent the luminous characters which compose VMS messages. The defined font includes, besides the alphanumeric characters, other 3D objects that model the pictograms that can be visualised on the graphic matrix.

As we will see later on, a whole pictogram is not assigned to a character of the 3D font, but to an element called subpictogram. A subpictogram is a one-colour graphic item that can commonly be part of different pictograms. By assigning subpictograms to characters instead of pictograms, the 3D font efficiency is enhanced, since there are several graphic elements which can be used to create numerous pictograms.

Therefore, we have not just created a font for the definition of the text messages, but also a font to define the pictograms related to the graphic matrix.

V. 3D MODELLING OF ALPHANUMERIC MATRICES

Each character of the 3D font that can be visualized on a VMS-panel corresponds to a certain luminous state of an alphanumeric matrix. This state makes reference to which pixels out of the 35 included in the matrix (7 row x 5 columns) are on and which ones are off. Depending on the pixels state (on or off), we can define a character or another.

A 3D object used to represent a character is composed of a set of square-shaped polygonal objects arranged in a suitable way called quads. Each one of these quads represents a pixel of the matrix.

Inside the user-defined 3D font that has been designed, each 3D model associated to a character is only made up of squares in amber that represents the pixels on. It is not necessary to include any other object in the character representation, neither the pixels off nor the dark area that surrounds the pixels of the matrix. The reason why is because in the virtual scene the objects that represent the characters are placed in front of a rectangular-shaped 3D object that simulates all the VMS-panel with its pixels off (Fig. 2(a)).

The character structure or, what is the same, the arrangement of the character pixels on is defined by means of a set of state masks. Each mask corresponds with a line of the alphanumeric matrix; therefore we have 7 masks per character. The mask of a row makes possible to define which pixels of the row are on and which are off. In the Table 1 we see an example of character as well as its definition by means of masks.

Another point to be taken into account is the character spacing. This property makes reference to the horizontal distance at which the next character should be drawn. Thereby, if the current character spacing is equal to zero, the next character should overlap it (like in the case of accent marks in romance languages). As we are not going to put accent marks in words, every character in the font will have the same spacing. Logically, the value of the spacing will be equivalent to the wide of the alphanumeric matrix plus the distance between each matrix.

Text messages on the VMS-panels are usually displayed in capital letters to achieve better legibility. Because of this, no lower-case letter has been assigned to any character of our font (Fig. 5). This fact has allowed utilizing the space reserved in the font to lower-case letters to associate 3D object representing the subpictograms of the graphic matrices.

VI. 3D MODELLING OF GRAPHIC MATRICES

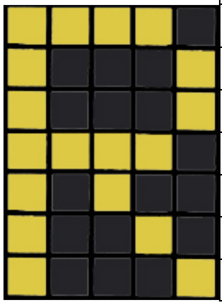
As we said before, each VMS-panel can have one or more graphic matrices, these matrices are frequently made up of 32 rows and 32 columns of pixels in five different colours: red, green, blue, white and amber. As we can see, there is a difference between this type of matrices and the alphanumeric ones: the graphic matrices can represent several colours whereas the others can only represent one colour. Therefore, we have to seek a different solution for their modelling.

Graphics matrices are basically used to visualise graphs or pictograms representing traffic code signs or others signs not included in the traffic code but following a similar philosophy. On the other hand, it is possible to represent by means of these matrices alphanumeric characters, although it will seldom be done.

We have considered two different alternatives for the definition of pictograms. These alternatives are:

1. Defining the pictograms by means of a 32x32-byte matrix, so that each byte contains the colour information of the pixels located in this position. The possible values are for example: 0 meaning off, 1 meaning red, 2 meaning green, 3 meaning blue, 4 meaning white and 5 meaning amber.

Table I – Design of the letter R and associated masks



Binary	Hexadecimal	Decimal
11110	1E	30
10001	11	17
10001	11	17
11110	1E	30
10100	14	20
10010	12	18
10001	11	17

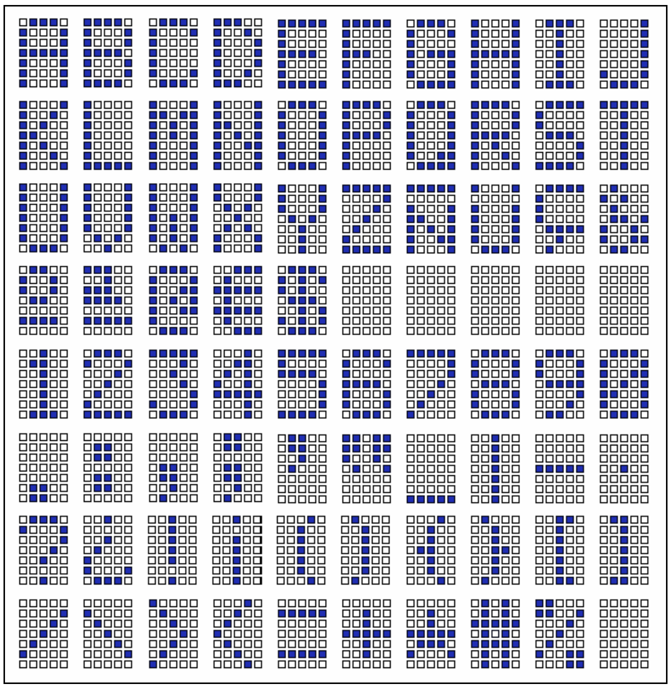


Fig. 5 – Alphanumeric character set of the 3D font

2. Dividing one pictogram into subpictrograms (Fig. 6). A subpictogram is a part or item of the pictogram that has the same colour. As examples of subpictograms we can quote: the red circle in the prohibition signs, the red triangle in the warning signs, arrows, etc.

We have chosen the second option to represent graphs because it has an important advantage: it considerably simplifies the definition of pictograms against the first alternative, due to the fact that most signs are composed of several elements that are usually repeated in other signs.

The definition of the subpictograms is performed in a similar way as in the case of characters, i.e., each row of pixels in a pictogram has a mask associated (32 masks altogether).

Furthermore, we must add to the subpictogram data its spacing and colour. Commonly, we have to specify that its spacing is equal to zero, so that, if an instance of the pfString class represents a string composed of several subpictograms, they will be drawn in the same position.

Let's see, for example, the case of the two subpictograms of the Fig. 6. Both icons must be located in the same position

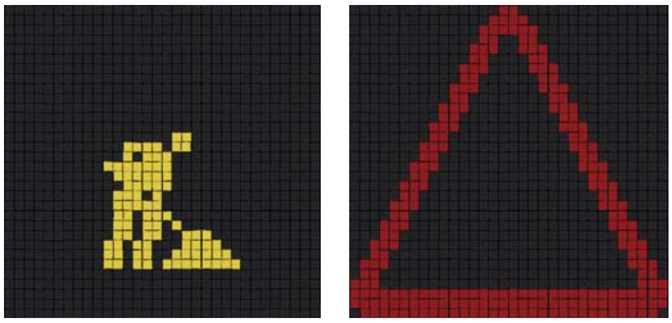


Fig. 6 – Subpictograms of the sign "Road Works"

to form the traffic code sign “Men at Work” correctly. This sign can be represented using our personalised 3D font by means of an object of the pfString having the string value “ah”, where the ‘a’ character represents the red triangle and the ‘h’ character represents the man working. Furthermore, in order to have both icons located in the same position, the subpictogram associated to the ‘a’ character must have a spacing equal to zero.

VII. DYNAMIC AND AUTOMATIC MESSAGES

As we have seen before, the definition of characters and pictograms by means of masks will allow constructing in a completely automatic way the 3D font of characters used to construct traffic messages. It is only necessary to do a previous work in order to define the shape of each item and calculate their related masks.

That is to say, initially the application defines a 3D font using the pfFont class and later, in run time, it will assign any desired message to a certain PMV model by means of the pfString and pfText classes. This assignation is dynamic, that means it is possible to change messages and pictograms at any time of the simulation execution.

VIII. CONCLUSION AND FUTURE WORK

In the current paper, we have exposed an architecture to define 3D models representing characters to construct the dynamics text messages. These messages can be shown in the PMV models in a real-time Driving Simulation System in order to evaluate them. Initially, some experimental tests have been performed to study several particularly interesting panel properties like comprehension, homogeneity, etc. Those first experiments have been carried out simulating ideal weather and visibility conditions. An extremely interesting future work could consist of carrying out tests with experimental subjects under adverse weather and visibility conditions (rain, fog, front sun, rear sun, etc).

IX. ACKNOWLEDGMENT

The author thanks his colleague Jean-François Pace for his help in proofreading this paper in order to check spelling.

REFERENCES

[1] Bayarri, S.; Pareja, I.; Coma, I. y Fernández, “M. Modelado de Carreteras para Simulación de Conducción”, 8º Encontro Português de Computação Gráfica. Coimbra, 1998.

[2] Pareja, I.; Coma, I.; Bayarri, S. y Rueda, S. “Modelado de calidad para la visualización interactiva de carreteras”, IX Congreso Español de Informática Gráfica. CEIG’99. Jaén, 1999.

[3] Sánchez, M.; Valero, P.; Coma, I.; Pareja, I.; Rueda, S. y Sanmartín, J. “Construcción de escenarios de conducción para el Sistema Informático de Evaluación Interactiva de Conductores por medio de Simulación (EVICA): La realidad no es siempre un buen modelo para la construcción de simuladores”, Congreso Internacional Sobre el Uso de Simuladores. León, 1999.

[4] Sánchez, M.; Valero, P. y Pareja, I.”Inferfaz de Usuario en el Desarrollo de un Simulador de Conducción”, Interacción 2000. I Jornadas de Interacción Persona-Ordenador. Granada, 2000.

[5] OpenGL Performer Programmer’s Guide. Silicon Graphics, Inc. 2002