Software de Autoría de Realidad Aumentada

4 de diciembre de 2010

Jesús Antonio Olán López[060300231@ucaribe.edu.mx]
Daniel Madrigal Reyes[060300222@ucaribe.edu.mx]
José Enrique Álvarez Estrada[jeae@ucaribe.edu.mx]
Universidad del Caribe
Cancún, Quintana Roo

Resumen

En este trabajo se realiza un intérprete para la librería ARtoolKit que permita realizar implementaciones de realidad aumentada a los no-programadores .El intérprete está programado en Pascal y recibe como único parámetro un archivo editado por el usuario con primitivas básicas que le permiten diseñar figuras en 3D, dotarlas de comportamientos y asociarlas a un marcador reconocido por ARtoolkit.

Palabras clave. Realidad aumentada, intérprete, ARtoolkit, OpenGL, Pascal

1. Introducción

Los sistemas con Realidad Virtual [1] que en los inicios de los 90's eran difíciles de desarrollar, hoy son una realidad, el crecimiento de la tecnología ha permitido su introducción en diversas áreas como el campo militar[2] o la psicología. La Realidad Aumentada es un término que se escucha, cada vez más en nuestro entorno, esto debido a las características con la que, los actuales dispositivos modernos cuentan. Por ejemplo, dispositivos como el IPhone de Apple o la Play Station Portable (PSP) de SONY [3] cuentan con aplicaciones de realidad aumentada.

La Realidad Aumentada difiere de la Realidad Virtual; mientras la Realidad Virtual se encarga de sustituir todo un entorno real, la Realidad Aumentada según [4], se encarga de combinar objetos virtuales (imágenes, texto, vídeo, audio, modelos 3D) con el mundo real. Existen bibliotecas de software disponibles para desarrollar aplicaciones con realidad aumentada, una de ellas es ARToolKit [5]. Un software open source, que nos permite modificar su código dependiendo

de nuestras necesidades, además de ser multiplataforma (Linux, Mac y Windows) extendiendo nuestras aplicaciones a otros S.O.

La realización de un proyecto con ARToolKit requiere de conocimiento de lenguajes de programación (Lenguaje C y OpenGL). La problemática se centra en personas que no cuentan con conocimientos de programación que desean crear sus propias aplicaciones en Artoolkit, por ejemplo: profesores [6] que deseen despertar el interés de los estudiantes con nuevos métodos para el proceso enseñanza/aprendizaje.

Entre las soluciones existentes encontramos ATOMIC Authoring Tools, la única aplicación multiplataforma y open-source que utiliza ARToolKit pero con limitaciones, ya que no permiten al usuario crear sus propias figuras, manipulación de las mismas, además de que no se pueden mostrar múltiples objetos virtuales. Para que los no-programadores tengan una interacción más directa a la hora de crear sus propias aplicaciones y así darles más libertad a la hora de crearlas debemos de lograr un enlace entre ARToolKit y el lenguaje del propio usuario. Por dicho motivo, nosotros proponemos realizar un software desarrollado en lenguaje pascal que interprete un archivo creado por el usuario para ARToolKit, mediante el uso de un lenguaje sencillo como lo es, la sintaxis declarativa.

2. Realidad Aumentada

El campo de la realidad aumentada existe hace poco más de una década, y desde entonces ha crecido notablemente. Su objetivo básico es mejorar la percepción del usuario y la interacción con el mundo real.

[4] define la Realidad Aumentada (RA) como la combinación de objetos y entornos 3- D, visualizados en tiempo real, que actúan como suplementos de la realidad, sin reemplazarla completamente.

También identifica tres características que debe tener un sistema AR:

1. Combinación de realidad y mundos virtuales.
2. Las imágenes virtuales se registran en el mundo real.
3. Interactividad en tiempo real.

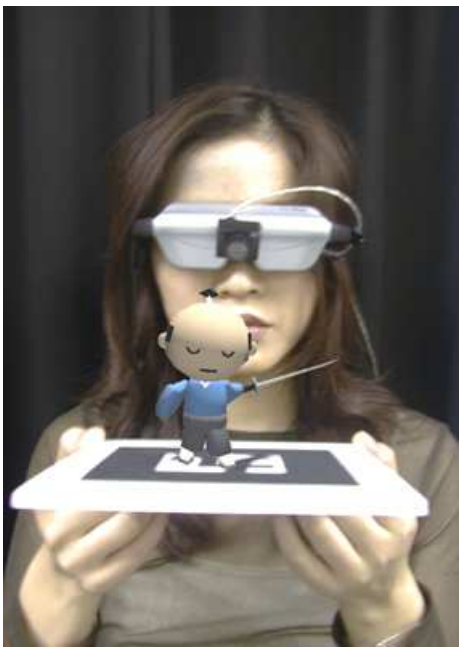


Figura 1: ARToolKit

La realidad aumentada puede extenderse a más sentidos, tales como el tacto, la audición, etc.Esta se capta mediante un sensor óptico (cámara), y se procesa digitalmente en busca de patrones previamente incrustados en el ambiente observado. Una vez reconocidos dichos patrones, se ubica su posición en el espacio, que se usa para agregar los objetos superpuestos a la salida de la imagen, creando la ilusión de que realmente están presentes en el mundo real.

2.1. ARToolKit

Es una biblioteca de software diseñada en lenguaje de programación C y C++ para la creación de aplicaciones de Realidad Aumentada (RA)(Figura. 1), fue creada por el DR. Hirokazu Kato en colaboración con M. Billinghurst. Dichas aplicaciones con realidad aumentada implican la superposición de imágenes virtuales en el mundo real. Lo que nos ofrece para desarrollar nuestras propias aplicaciones, es lo siguiente:

1. Un Framework para crear aplicaciones de realidad aumentada en tiempo real.

2. Una librería multiplataforma (Linux, Mac OS x,Windows, SGI).

3. Superposición de objetos virtuales en 3D sobre marcadores reales (basado en un algoritmo de visión por computadora).

4. Una plataforma multivideoteca con :
    - Múltiples fuentes de entrada (USB, Firewire, tarjeta de captura).
    - Múltiples formatos (RGB/YUV420P, YUV).
    - Seguimiento de múltiples cámaras.
5. Seguidor de marcadores para patrones.
6. Una rutina de calibración.
7. Soporte para 3D VRML.
8. Una API en el lenguaje C.
9. Otros lenguajes soportados (Java, MATLAB).
10. Es OpenSource.

2.2. Patrón

El patrón es una marca con características únicas, que debe reconocer el módulo de procesamiento de imágenes de ARToolKit, cada vez que éste, se encuentre dentro de su espacio de visión. Estos marcadores se utilizan debido a que ARToolKit necesita saber en qué coordenadas del mundo real debe superponer el objeto virtual que el usuario escogió.

2.3. Reconocimiento de Patrones

El reconocimiento de patrones es la disciplina científica que estudia los métodos de descripción y clasificación de objetos [8], consiste en la aplicación de ciertas técnicas matemáticas, a señales que se obtienen a partir de procesos de segmentación, extracción de características y descripción, donde cada objeto queda representado por una colección de descriptores que los diferencian. Se puede resumir este proceso de la siguiente forma:

1. Adquisición de datos.
2. Extracción de características.
3. Toma de decisiones a partir de las características.

3. OpenGL

OpenGL[1] es una interfaz de software para hardware de gráficos. Esta interfaz consiste en cerca de 150 comandos que se utilizan para especificar los objetos y operaciones para producir una aplicación interactiva en 3D. Con OpenGL se pueden crear figuras 3D como piezas de automóviles o moléculas partiendo de primitivas geométricas como puntos, lineas y polígonos.

4. Lenguaje de Programación Pascal

El lenguaje de programación PASCAL fue desarrollado por Niklaus Wirth en los 70's, en la Universidad Técnica de Zurich, Suiza. El propósito fue crear un lenguaje de alto nivel para enseñar programación estructurada. El nombre PASCAL fue elegido en honor de Blaise Pascal (1623-1662), brillante científico y matemático francés.

PASCAL es un lenguaje cercano a los lenguajes y procesos de pensamiento humanos; ha sido diseñado para estimular el uso de la "programación estructurada",en un enfoque ordenado y disciplinado de la programación que conduce a la obtención de programas claros, eficientes y libres de errores. Sus instrucciones o sentencias se componen de expresiones de apariencia algebraica y palabras reservadas en ingles como BEGIN, END, READ, WRITE, IF, THEN, REPEAT, WHILE, DO.

[1] véase www.opengl.com

5. Lenguaje Declarativo

Los lenguajes declarativos se basan en la premisa de qué llevar acabo; es decir, el usuario indica que necesita y el computador regresa una respuesta. Estos lenguajes tienen la ventaja de contener pocas instrucciones, siendo su sintaxis fácil de aprender, pero estableciendo un limite en el propio lenguaje.

6. Propuesta

6.1. Interpréte

El intérprete es un programa informático capaz de analizar y ejecutar otros programas escritos en un lenguaje de alto nivel. Una diferencia entre los intérpretes y los compiladores, es que los compiladores traducen un programa de un lenguaje de programación al código máquina del sistema, los intérpretes realizan la traducción instrucción por instrucción y al terminar su ejecución no guardan el resultado de la traducción que realizaron.

Los programas interpretados pueden ser más lentos, esto debido a la traducción durante la ejecución del programa; sin embargo, son más flexibles como entornos de programación. Esta sencillez de código fuente permite añadir con gran facilidad nuevos módulos al programa, y le ofrece al programa interpretado un entorno que no depende de la máquina donde se ejecuta el intérprete, esto se conoce comúnmente como máquina virtual.

Decidimos utilizar un intérprete para crear el software de autoría de realidad aumentada ya que nos permite que un archivo con código fuente pueda funcionar en plataformas diferentes. La simplicidad del lenguaje declarativo que contendrá el código fuente es la principal característica por la que el intérprete, esta enfocado principalmente a personas sin conocimientos de programación.

6.2. Analizador Léxico

El analizador léxico(Scanner) recorre los caracteres del archivo .ARDJ hasta reconocer un Token(Componentes Léxicos), para ello habrá saltado espacios y comentarios que pudieran haber delante del Token. Un Token es una unidad léxica indivisible como son las palabras reservadas propias del lenguaje definido por nosotros. Un token nos permite saber si es:

- Un Identificador
- Una Palabra Reservada
- Un Entero Etc.

Figura 2: Ejemplo Archivo.ardj

6.3. Analizador Sintáctico

El analizador sintáctico(Parser), tiene la misión de solicitar Tokens al Scanner para reconocer frases. El parser reconocerá errores de sintaxis cometidos por el usuario en el archivo .ARDJ y le informará de ellos al usuario. Cuando el proceso del Analizador Sintáctico se lleva a cabo correctamente, es posible trabajar con la parte correspondiente de ARToolkit.

7. Desarrollo

7.1. Intérprete y ARToolkit

ARToolkit necesita que le indiquemos que objetos, marcadores y comportamientos deseamos; todo por medio de código en lenguaje de programación C y OpenGL, esto es complejo para las personas que no conocen la programación, aquí es entra el interprete, llevara acabo todos esos procesos traduciendo del archivo .ARDJ el código fuente a Lenguaje C y OpenGL. Pero para llevar acabo los procesos antes mencionados tenemos que hacer modificaciones en ARToolKit, debido a que ARToolKit esta en lenguaje C y el intérprete en lenguaje PASCAL.

7.2. H2PAS

H2PAS un comando que nos permite pasar archivos .h a .pp, esta es la parte más importante del desarrollo, aqui el comando H2PAS tratará de convertir fielmente las librerías, por ello tenemos que verificar que cada archivo .pp, tenga las definiciones de funciones, procedimientos y variables correctas.En este proceso tuvimos que modificar uno macro y algunos tipos de variables definidos en param.pp.

En todas las librerías que convertimos tenemos que hacer eso de la directiva PACKRECORDS C, para el alineamiento de de bytes que maneja el lenguaje C.

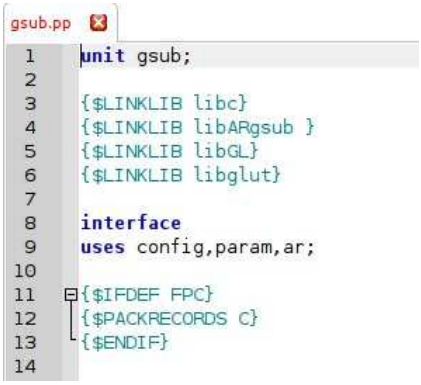


Figura 3: Enlazando librerías de ARToolKit Externas y Empaquetadas

7.3. Librerías ARToolKit a Pascal

Para lograr crear un programa en pascal con ARToolKit, necesitamos usar las funciones de ARToolkit en lenguaje de programación pascal, la forma de lograrlo es mediante los archivos .h de ARToolKit, estos contienen los prototipos de funciones y variables que necesitamos. Localizamos los .h que necesitamos basándonos de un programa ejemplo que viene con ARToolKit: loadMultiple.c

gsub.h: Contiene las principales funciones de OpenGL para visualización en ARToolKit.

param.h: Contiene las rutinas para cargar ,guardar y modificar los parámetros de la cámara.

ar.h: Contiene las funciones primordiales de ARToolKit ,rutinas de análisis de imagen, detección de marcadores, configuración de la cámara, entre otras.

video.h: Provee el soporte de entrada de video multi-plataforma.

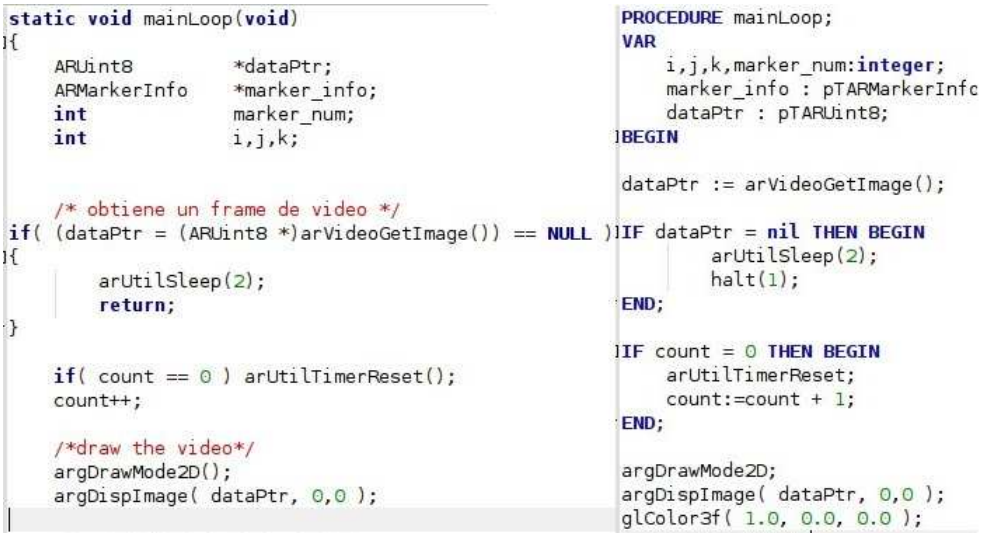
object.h: Contiene la definición de la estructura del marcador y una función parse que lee un archivo que contiene los patrones a usar.

7.4. Librerías empaquetadas de ARToolKit

Al igual que las librerías externas ARToolkit contiene un conjunto de empaquetados .a, son librerias que nos permitiran hacer un uso adecuado de funciones y argumentos de ARToolKit, es necesario agregarlas a en cada una de las .pp(gsub.pp, param.pp, ar.pp, video.pp)con la directiva LINKLIB.

7.5. Librerías Externas

Algunas de las librerías .pp(gsub.pp, param.pp, ar.pp, video.pp) necesitan de otras librerias externas para poder funcionar estas son: GSTREAMER (Mul-



```
static void mainLoop(void)
{
    ARUint8         *dataPtr;
    ARMarkerInfo    *marker_info;
    int             marker_num;
    int             i,j,k;

    /* obtiene un frame de video */
if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL )
{
        arUtilSleep(2);
        return;
}

    if( count == 0 ) arUtilTimerReset();
    count++;

    /*draw the video*/
    argDrawMode2D();
    argDispImage( dataPtr, 0,0 );
```

```
PROCEDURE mainLoop;
VAR
    i,j,k,marker_num:integer;
    marker_info : pTARMarkerInfo
    dataPtr : pTARUint8;
BEGIN

dataPtr := arVideoGetImage();

IF dataPtr = nil THEN BEGIN
        arUtilSleep(2);
        halt(1);
END;

IF count = 0 THEN BEGIN
    arUtilTimerReset;
    count:=count + 1;
END;

argDrawMode2D;
argDispImage( dataPtr, 0,0 );
glColor3f( 1.0, 0.0, 0.0 );
```

Figura 4: ARToolKit: Código en C y Código en Pascal

timedia), C (Lenguaje C), GL (OpenGL), GLUT (OpenGL), gstvideo4linux (Video), gstvideo4linux2 (Video). Dentro de pascal estas se enlazan con la directiva LINKLIB para hacer uso de ellas[9].

7.6. ld.so.conf

Este archivo es la configuracion de un demonio que carga librerias que se encuentra en linux(/etc/ld.so.conf), aqui se incluyo la ruta de la libreria GSTREAMER para hacer uso de ella.Despues con el comando ldconfig actualizamos la librerias del S.O.

7.7. Implementando un Ejemplo

Para realizar las pruebas de las funciones y procedimientos básicos de ARToolKit que ya estan en Pascal, se escribió un código similar a uno de los ejemplos que estan en ARToolKit pero en lenguaje C(Figura. 4), con esto llevamos a prueba el despliegue de video y la ejecución correcta del programa, lo cual lógicamente representara que las librerias hasta el momento funcionan.

8. Resultados

Los resultados que obtuvimos fueron:

- Archivo .ARDJ con sintaxis de tipo declarativa.
- Unit en Pascal de ARToolKit.
- Intérprete, el cual nos permite una traducción de las primitivas más basicas de ARToolKit además de asociarlas a un Marcador.
- El uso de Múltiples Marcadores.
- Generar una aplicación básica de RA

9. Conclusiones

La obtención de una unit de ARToolkit en Pascal fue la columna vertebral del proyecto, esto nos permitió que siguieramos el desarrollo del intérprete en pascal, lo que obtuvimos al final de todos estos procesos, fue una unit funcional. Todavía quedan detalles que se pueden mejorar, parámetros en funciones que pueden referenciarse de mejor manera, lo que podría ayudar al rendimiento de ARToolKit. Existen también funciones y procedimientos de ARToolKit en los que se necesita profundizar para comprender más sus procesos internos, esto nos puede permitir mejorar o aprovechar más las utilidades con las que cuenta y que no usamos para este proyecto.

El intérprete nos permitió crear aplicaciones básicas, con ello comprobamos que la inclusión de una aplicación sobre ARToolKit es factible, podemos mejorar la unión entre estos dos módulos, por medio del uso de estructuras dinámicas; claro esto nos llevaría a volver más complejo el intérprete, pero también mucho más potente, esto al parecer es la linea a seguir en un futuro.

La estructura que definimos para el archivo.ardj implícitamente y sin saberlo, nos lleva a lo que se conoce como framework, es decir; la programación que constituye a intérprete, se puede programar orientado a objetos, debido a que las figuras complejas podrían tener formas de manipulación es decir eventos.

Se pusieron en práctica aquellos conocimientos adquiridos en las materias de Compiladores y Sistemas Operativos POSIX lo cual fue clave para poder llevar a cabo el desarrollo de este proyecto, el conocimiento nuevo, hasta éste momento adquirido abre la posibilidad de que otros programas escritos exclusivos en lenguaje de programación C, puedan ser exportados a Pascal.

Referencias

[1] Chan Zhao, *Evaluation of VRML for Modeling Virtual Worlds*, (2000)

[2] bichotoblog.com/2010/06/17/realidad-aumentada/

[3] InviZimals *http://es.playstation.com/psp/games/detail/item286412/InviZimals-La-Otra-Dimensión/* (2010)

[4] Azuma Ronald T., *Fundamentals of wearable computers and augumented reality*, (2001)

[5] ARtoolKitDr. Hirokazu Kato *http://www.hitl.washington.edu/artoolkit/*, (1999)

[6] Valverde Andreu, Juan Antonio, *Compiladores e intérpretes: un enfoque pragmático*, (1989)

[7] *www.universia.edu.pe/noticias/principales/destacada.php?id=7555*

[8] Marques de Sá Joaquim P., *Pattern Recognition: concepts, methods, and applications*, (2001)

[9] G. Marcou, E. Engler, A. Varnek, *How to use C code in Free Pascal projects*

[10] Milgram Paul, Kishino Fumio, *A Taxonomy of reality mixed visual displays*, (1994)

[11] R. Silva, J. C. Oliveira, G. A. Giraldi, *Introduction to Augmented Reality*, (2003)