JavaVis: Una librería para visión artificial en Java

Miguel Cazorla, Otto Colomina, Patricia Compañ, Francisco Escolano, José L. Zamora

Dept. de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
Apartado 99, 03080 Alicante
e-mail: {miguel,otto,patricia,sco}@dccia.ua.es

Resumen

En este artículo se describe *JavaVis*, una librería de visión artificial escrita en Java y desarrollada en el Departamento de Ciencia de la Computación e Inteligencia Artificial de la Universidad de Alicante. Esta librería presenta una serie de ventajas: portabilidad, fácil ampliación, interfaz gráfico y formato de imagen especial que permite almacenar secuencias de imágenes y datos de tipo geométrico. La librería se utiliza en las prácticas de Visión Artificial que se vienen impartiendo en las asignaturas de I.A. de la Universidad de Alicante.

1. Introducción

En este artículo se presenta la librería de visión artificial JavaVis [1], desarrollada en el Departamento de Ciencia de la Computación e Inteligencia Artificial de la Universidad de Alicante.

En esta universidad se imparten actualmente diversas materias de Inteligencia Artificial cuyo temario incluye conceptos de Visión Artificial [2], concretamente, en la Ingeniería Informática, las asignaturas de Técnicas de I.A. y Ampliación de I.A., y en el programa de Doctorado la asignatura de Visión Tridimensional. Para las prácticas de visión de dichas asignaturas se venía utilizando hasta el momento la librería VISTA [3].

La ventaja principal de Vista es su filosofía de programación, ya que es posible añadir nuevos algoritmos de manera sencilla, e incorpora un formato especial de imágenes que permite, por ejemplo, almacenar secuencias de imágenes, datos geométricos, etc. en el mismo fichero de imagen. Incluso el usuario puede definir sus propios tipos de datos para incluirlos en la imagen. No obstante, también tiene inconvenientes: al estar desarrollada en C bajo Unix/Linux su portabilidad es limitada, y aunque se puede manejar perfectamente mediante línea de comandos, su interfaz gráfico es demasiado pobre. Esto nos llevó a la idea de desarrollar una librería que incorporara las características más interesantes de Vista (formato de imágenes, filosofía de programación), aunque eliminando sus puntos débiles (portabilidad e interfaz de usuario). Con el objetivo de mejorar en estos dos últimos aspectos, se decidió realizar la implementación de la librería en el lenguaje Java [4].

En los siguientes apartados se describen las características principales de JavaVis. En el apartado 2 se detallan los objetivos perseguidos a la hora de desarrollar la librería y el formato de imagen que emplea ésta. En el apartado 3 se detallan las funciones de visión artificial y procesamiento de imágenes que incorpora actualmente la librería. Finalmente se recogen unas breves conclusiones.

2. La librería JavaVis

Esta librería se empieza a desarrollar en el curso 1998/99 como un trabajo de la asignatura Sistemas Informáticos de la Ingeniería en Informática en la Universidad de Alicante. En dicho curso se implementan las clases básicas para la lectura de fichero, manejo de imágenes y acceso a píxeles. En el siguiente curso académico se incorporan las clases para el manejo del tipo de datos geométrico. También se desarrollan varios algoritmos clásicos de visión artificial, como el

algoritmo de Canny de detección de aristas. En el presente curso académico se siguen incorporando algoritmos, fundamentalmente de visión estéreo. En las siguientes secciones se detallan los objetivos que nos planteamos al desarrollar esta librería así como el formato de imagen especial desarrollado.

2.1 Objetivos

Los objetivos que se perseguían en el desarrollo de esta librería de procesamiento de imágenes fueron los siguientes:

1. Utilización de un lenguaje de programación multiplataforma, orientado a objetos y de aplicación en la enseñanza. El lenguaje Java incorpora todo ello y además permite una integración inmediata en Internet.

2. Abstracción de la interfaz gráfica. Este objetivo pretendía que el programador se olvidara de comandos de manejo del entorno. En esta librería simplemente tenemos que conocer el lenguaje Java y dedicarnos a desarrollar nuestro algoritmo. Después dejamos el fichero que contiene el algoritmo en un directorio determinado y la librería se encarga de hacerlo visible para el usuario. La Figura 1 muestra un ejemplo de ejecución de la interfaz gráfica desarrollada.

3. Abstracción del paso de parámetros y lectura de ficheros. De igual manera que el anterior objetivo, el paso de parámetros y lectura de ficheros los maneja la librería. El programador no tiene que preocuparse por las clases o métodos que manejan los parámetros de cada algoritmo. Estos parámetros cambiarán para cada algoritmo: en el caso del ajuste de brillo será un entero que indica la cantidad o porcentaje de brillo que queremos aumentar o disminuir y, por ejemplo, en el caso del algoritmo de Canny son un número real (la varianza) y un entero (cantidad de brillo). En la librería nos evitamos el tener que ir pidiendo los parámetros

4. Utilización desde entorno gráfico, línea de comandos o bien llamada desde otro algoritmo. Las funciones implementadas pueden ser utilizadas desde un entorno gráfico integrado en la librería. Este entorno permite un uso para aquellos usuarios que no deseen conocer el funcionamiento exacto de estos algoritmos. Otra forma de utilización es desde línea de comandos. Desde línea de comandos del sistema operativo podemos llamar a los algoritmos implementados pasando como parámetros ficheros con imágenes. Por último, cada algoritmo permite ser llamado por otro algoritmo. Por ejemplo, el algoritmo de Canny puede ser llamado por otros algoritmos que necesiten una detección de aristas como parte de su desarrollo.

5. Manejo de un tipo especial de ficheros e imágenes. La mayoría de formatos de imágenes (jpg, tif, bmp, etc.) sólo permiten una imagen por fichero. Sin embargo, los algoritmos de visión artificial muy a menudo necesitan más de una imagen por ficheros. Pongamos un ejemplo: si estamos calculando el flujo óptico de una secuencia de imágenes lo ideal sería disponer de todas las imágenes en un mismo fichero o secuencia. Además, si disponemos de visión estéreo en cada instante tendremos dos imágenes, una correspondiente a la imagen izquierda y otra a la derecha (bandas). Tanto la secuencia como las bandas se contemplan en esta librería.

2.2 Tipo de datos imagen

Como hemos comentado previamente una imagen esta compuesta por una secuencia de *frames.* Cada *frame* está compuesto a su vez por una o varias bandas (1, 2, 3, …). Cada banda es ya una imagen. Distintos *frames* en una misma secuencia pueden contener distinto número de bandas. Un fichero contendrá una secuencia. Cuando aplicamos un algoritmo a un fichero, bien desde línea de comandos o desde el entorno, el algoritmo se aplica a toda la secuencia y el resultado es otra secuencia con el mismo número de *frames* que la secuencia original y donde a cada *frame* se le ha aplicado el algoritmo.

3. Funciones desarrolladas

A continuación se detallan las funciones disponibles en la librería:

- Conversión: Convierten los formatos de las imágenes. *FColorToGray*, *FGrayToGray*

- Ajustes: Realizan ajustes de brillo, contraste e intensidad en la imagen. *FBrightness, FConstrast, FGamma, FEqualize.*
- Suavizado: Realizan operaciones de suavizado para reducir ruido y otros efectos que pueden estar presentes en la imagen. *FSmoothAverage, FSmoothMedian, FSmoothGaussian, FSmoothRadial.*
- Realce: Efectúan operaciones de realce sobre la imagen para compensar efectos tales como sombras y falsos reflejos. *FSharpen, FSharpenMore.*
- Convolución: Aplican una convolución a la imagen con una máscara o con otra imagen.. *FUser3x3, FUser5x5, FConvolveImage, FConvolveAscii.*
- Manipulación: Realizan operaciones tales como escalado, giro, rotación, deformación de la imagen, etc. *FSkew, FFlip, FMirror, FScale, FRotate, FWaveHoriz, FWaveVert, FNegate, FNoise.*
- Geometría: Añaden puntos, segmentos a la imagen y convierten una imagen de tipo geométrico a escala de grises. *FAddPoint, FAddSegment, FRandomPoint, FRandomSegment, FInterSegment, FGeoToGray.*
- Contorno: Realizan detección de contornos. *FCanny, FLink, FSegEdges.*
- Gradiente: Calculan el gradiente de la imagen. *FGrad, FMag, FPhase.*
- Otras: en esta categoría se incluyen las funciones creadas por el usuario. La librería incorpora las siguientes: *FHistogram, FSkeleton, FBinarize, FCrop, FMaximum, FMinimum, FPixelate, Fop.*

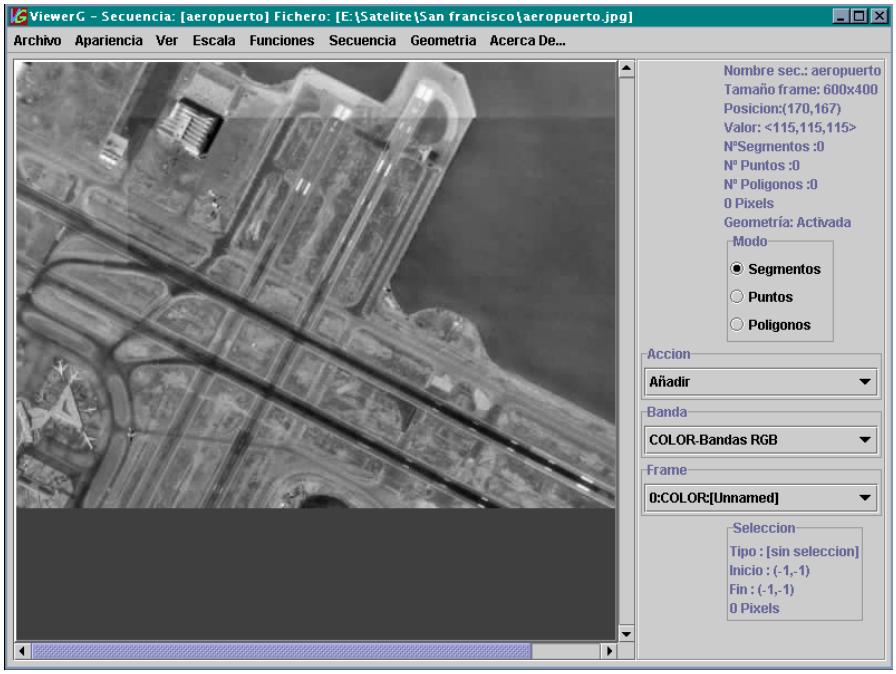

Figura 1: Entorno de la librería JavaVis.

4. Conclusión

Se ha presentado la librería de visión artificial JavaVis, que ofrece una serie de ventajas con respecto a otras librerías: portabilidad, posibilidad de incorporar nuevos algoritmos de manera

sencilla, formatos de imagen mejorados y abstracción con respecto al interfaz gráfico.

La librería se ha comenzado a utilizar en el curso 2000/2001 en las asignaturas de Técnicas de I.A. y Ampliación de I.A, de la Ingeniería en Informática, así como en la asignatura de Visión Tridimensional del programa de Doctorado en Ingeniería Informática y Computación.

5. Agradecimientos

Esta librería no se hubiera desarrollado sin el esfuerzo de los alumnos de la asignatura de sistemas informáticos. A ellos, a su conocimiento casi ilimitado del lenguaje Java y a su empeño en sacar adelante algunas de las ideas propuestas estamos profundamente agradecidos. La librería empezaron a desarrollarla Juan Pablo García Calderón y Mª Carmen Hernández Rosa y la continuaron Juan Carlos García Candela, Alberto Francisco Luciáñez Belda y Francisco Manuel Serrano González.

Referencias

[1] Página Web de la librería: http://www.dccia.ua.es/~miguel/JavaVis/start.htm

[2] Trucco & Verri. *Introductory Techniques for 3D Computer Vision*, Prentice-Hall, 1998.

[3] M.A. Cazorla. La librería Vista como una herramienta de investigación. Technical Report DTIC-1996-II-4, Departamento de Tecnología Informática y Computación, Universidad de Alicante, junio 1996.

[4] M. Campione et al., *The Java Tutorial*, Addison-Wesley, 2000.