Documentación de Descartes 5

Alejandro Radillo Díaz
José Luis Abreu León
Joel Espinosa Longi

1 de septiembre de 2016

Índice general

Capítulo 1

Sobre la presente documentación

La presente documentación está destinada tanto a personas que no han usado Descartes como a personas que tienen cierta experiencia y desean mejorarla.

La documentación aborda tanto al Editor de Descartes 5 como la funcionalidad general del mismo. No obstante, se da una importancia mayor a la funcionalidad. Adicionalmente, se trata la funcionalidad más común de Descartes, y no se abordan algunas particularidades particulares del Editor y algunas de funcionalidad general debido a que no son de uso frecuente.

Con el objeto de que el lector pueda practicar a medida que avanza en la documentación, se proponen algunos ejercicios cuando se considera se ha acumulado suficiente conocimiento para ejecutarlos, y se incluyen los ejercicios resueltos en forma de interactivos de Descartes para que el lector pueda compararlos con los propios. Se encuentran en la carpeta *Ejercicios* dentro del archivo comprimido *DocumentacionDescartes.zip* en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/. Dentro de esta documentación se incluyen los mismos interactivos **en la red**, pero es importante recordar que están presentes en el zip proporcionado para poder revisarlos de manera local. Asimismo, es importante mencionar que los vínulos en el documento que redirigen al ejercicio en la red, redirigen a un archivo web que permite tanto visitar el interactivo terminado como la serie de instrucciones para lograrlo. Cabe mencionar que existen algunos comandos en las escenas de los interactivos que no vienen en las instrucciones particulares del interactivo y están ahí sólo para que el interactivo pueda ser visualizado correctamente en el contenedor en donde el usuario puede alternar entre el interactivo completo y el documento con las instrucciones.

Los ejercicios típicamente consisten en una serie de pasos guiando al usuario a crear un interactivo. No obstante, de forma intencional se busca que las instrucciones de los pasos no sean explícitas. Ello favorecerá que el lector pueda practicar, experimentar abordajes distintos, y revise, de ser necesario, la documentación para lograr una escena final. En este espíritu, es posible que un interactivo diseñado por el usuario difiera del proporcionado en la documentación. Dos estrategias diferentes pueden llevar a un mismo resultado. Adicionalmente, se proporcionan observaciones a cada paso. Normalmente, éstas observaciones mencionan el resultado esperado de cada paso, pero algunas ocasiones, éstas podrán traer sugerencias o información que ayude al usuario a lograr el paso correspondiente.

Los ejercicios en este documento van aumentando de dificultad. Poco a poco involucran más y más compleja funcionalidad. Por lo mismo, es buena idea que se en la medida de lo posible se aborde en orden, aunque existe la posibilidad de visitar la funcionalidad diversa de Descartes mediante los vínculos proporcionados.

Con esta documentación se espera que el usuario pueda, al concluir esta documentación, generar sus propios interactivos con Descartes 5. Y gracias a la versatilidad de esta herramienta de programación,

es probable que el usuario termine generando escenas de mayor complejidad a las presentadas a lo largo de este documento.

Es preciso mencionar que esta documentación no es estática. Se pretende agregarle mejoras con el tiempo, por lo que se sugiere descargar nuevas versiones de la misma con regularidad.

Capítulo 2

¿Qué es Descartes 5?

2.1. Los inicios de Descartes

Descartes nace a fines del siglo XX como una herramienta de creación de interactivos que aprovecha el lenguaje de programación Java. Consiste en un programa en Java que permite generar archivos .html para ser visualizados como páginas web. Los archivos html generados por Descartes suelen contener interactivos y son principalmente usados en la docencia de matemáticas y física en diversos niveles.

A pesar de que existe una gran variedad de programas interactivos con fines de docencia tales como GeoGebra, Cabri y otros, Descartes permite una gran versatilidad de interacciones. Adicionalmente, con Descartes es posible generar interacciones específicas diseñadas por el profesor, ejercicios aleatorios con restricciones particulares, además de que presenta muchas otras ventajas como se verá adelante.

En sus inicios, *Descartes* funcionaba directamente en Java para ser usados en computadora. No obstante, el advenimiento de dos tecnologías nuevas forzaron un cambio en Descartes, a saber:

- El advenimiento de dispositivos móviles
- El uso de JavaScript y HTML5

2.2. Modificaciones actuales

Se volvió entonces necesario que los interactivos corrieran en dispositivos móviles (en JavaScript). Descartes 5 nace de esta necesidad. Un nuevo editor fue creado con esto en mente. Aunque no hubieron cambios muy drásticos en el editor, fue necesario incluir una librería nueva. Todo esto se explica un poco más a fondo adelante.

Capítulo 3

Introducción a los componentes de Descartes 5

Descartes 5 contiene dos elementos principales, a saber:

1. El editor
2. La librería

3.1. El editor

El editor consiste en el programa en java (*Descartes.jar*). Este archivo puede bajarse de la red en este vínculo: http://arquimedes.matem.unam.mx/Descartes5/distribucion/Descartes.jar. Dicho archivo se encuentra en constante mantenimiento incluyendo nuevas mejoras, de tal forma que es siempre una buena práctica bajarlo con cierta frecuencia (en mi experiencia, cada mes).

El programa del editor es una interfaz gráfica para el usuario mediante la cual podrá guardar archivos html con diversos tipos de interactivos. En la Figura 3.1 se muestra la vista del editor, y el uso del mismo se detalla más adelante.

Recomendación: El programa *Descartes.jar* puede guardarse en cualquier directorio. De preferencia, se recomienda uno cercano a la raíz del sistema. Por ejemplo, en Windows podría guardarse en un directorio creado por el usuario *c:/Descartes*.

Cuando el programa es ejecutado, se generan en su carpeta contenedora dos archivos adicionales. El primero, *Descartes.bat*, es un archivo ejecutable. Su contenido es el texto '@java -cp Descartes.jar -ms64M -mx256M Descartes `%1 %2 %3`'. Este archivo puede modificarse cambiando el texto 'Descartes.jar' por '*c:/Descartes/Descartes.jar*' y puesto en cualquier carpeta. Al ejecutarlo, permitirá usar el editor directamente en la carpeta elegida, siempre y cuando el archivo jar esté guardado en '*c:/Descartes/*'.

3.2. La librería

La librería es un archivo js que se puede bajar del siguiente vínculo: http://arquimedes.matem.unam.mx/Descartes5/lib/descartes-min.js. Sin embargo, cada vez que el editor se abre, permite realizar una actualización automática de la librería, como se muestra en la Figura 3.2. El diálogo de actualización de las librerías sólo aparece si se detecta que las librerías están desactualizadas.

Al archivo *descartes-min.js* también se le conocerá de ahora en adelante como el ***intérprete***, debido a que es un archivo que permite a los navegadores entender el código de un archivo html creado por el

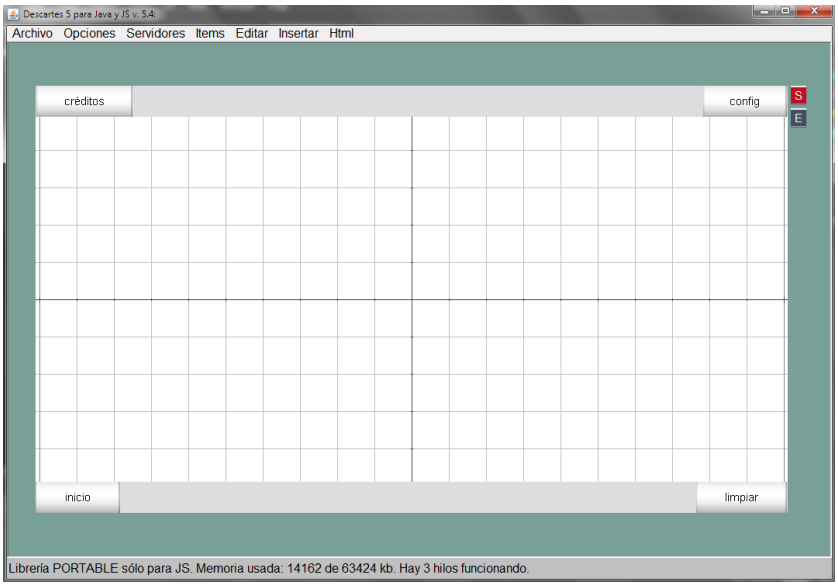

Figura 3.1: El editor de *Descartes 5* como se abre originalmente.

Figura 3.2: Ventana de actualización de librerías.

Editor de Descartes para así desplegarlo correctamente. Se recomienda siempre aceptar la actualización para que los interactivos sean visualizados correctamente.

Capítulo 4

Aprendiendo a usar el editor

4.1. Lanzando el editor desde una ventana de comandos

Como se vio anteriormente, el editor puede lanzarse ejecutando directamente el archivo *Descartes.jar* o mediante una ventana de comandos (lanzando el archivo Descartes.bat mencionado anteriormente).

En la Figura 4.1 se puede observar la ventana de comandos tras lanzar el archivo *Descartes.bat*.

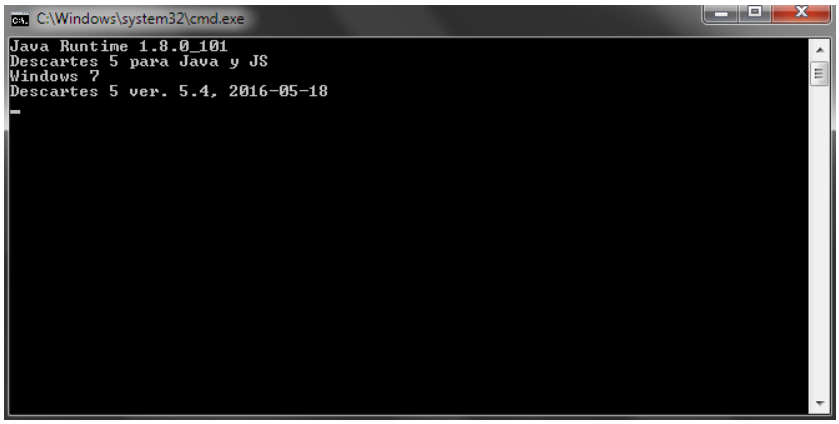

Figura 4.1: Ventana de comandos desplegada al lanzar el archivo *Descartes.bat*.

Al ejecutar Descartes lanzando el *Descartes.bat*, no sólo se abre el editor, sino una ventana de comandos. Ésta ventana tiene las siguientes virtudes:

1. Si en algún momento el editor se traba o cuelga por algún ciclo mal programado, se puede cerrar el editor cerrando simplemente la ventana de comandos. De forma general, el cerrar esta ventana siempre cerrará el Editor.
2. En caso de haber errores en la programación, la ventana de comandos siempre los despliega. En muchos casos, el editor por su cuenta sólo muestra los errores la primera vez que se los encuentra por medio de una ventana emergente. Una vez que han sido mostrados, si el mismo error ocurre de nuevo, el editor no lo mostrará. Sin embargo, la ventana de comando **siempre** muestra los errores. Esto es útil cuando se busca corregir algún programa, ya que si la ventana de comandos deja de desplegar el error, quiere decir que éste ha sido corregido. Se verán ejemplos detallados de esto más adelante.

3. Supongamos que deseamos trabajar en la carpeta *c:/MisDescartes/* y tenemos el archivo del programa en *c:/Descartes/*. Conviene colocar el archivo *Descartes.bat*, cuyo contenido en texto es '@java -cp c:/Descartes/Descartes.jar -ms64M -mx256M Descartes `%1 %2 %3`', en la carpeta *c:/MisDescartes/*. Al ejecutar el archivo .bat, el editor se abrirá, y todos los comandos del menú (tales como Abrir, Guardar como, etc.) siempre buscarán en *c:/MisDescartes/*. De tal suerte que no será necesario estar cambiando de directorio.

Es importante tener en cuenta que en sistemas con Linux, será necesario lanzar el archivo *Descartes.jar* a partir de una consola. La consola sirve en Linux como la ventana de comandos en Windows.

4.2. La barra de menús del Editor

Al igual que cualquier programa, el Editor tiene una barra de menús. A continuación se detallan los dos primeros, que son los más comúnmente usados:

4.2.1. El menú *Archivo*

- Nuevo: Crea un documento de Descartes en blanco.
- Abrir: Permite abrir un documento html creado en Descartes.
- Cerrar: Cierra el documento actual de Descartes.
- Actualizar: Actualiza el interactivo con los cambios más recientes.
- Guardar: Si aún no había sido guardado el html, se muestra un diálogo para escoger el nombre y ruta de guardado. De lo contrario, se sobreescribe el archivo con los nuevos cambios. Guardar de forma periódica es una costumbre sabia para no perder el progreso en caso que la herramienta llegara a colgarse.
- Guardar como: abre un diálogo para escoger ruta y nombre de un nuevo archivo html en el que se guardarán la última versión del mismo.
- Exportar a png / Exportar a jpeg: abren respectivamente un diálogo en el que se puede escoger la ruta y nombre de una versión gráfica del archivo html (ya sea en formato png o jpg). Suelen usarse para generar íconos de unidades o escenas hechas en Descartes como vista previa.

4.2.2. El menú *Opciones*

Este menú contiene opciones relacionadas a la librería (el archivo descartes-min.js discutido antes). Cada elemento del menú tiene dos opciones: *para Java y JS* y *sólo para JS*. *JS* en este caso se refiere a *JavaScript*. La opción *para Java y JS* solía usarse cuando los interactivos también corrían en Java. No obstante, dado que ahora los interactivos corren principalmente en JS, esa opción se ha vuelto obsoleta. Es incluso probable que deje de aparecer en versiones futuras del editor.

A continuación se analizan las tres opciones principales de este menú:

- Librería en Internet: Cuando se elige alguna opción de guardado del menú *Archivo*, se lee el archivo *descartes-min.js* de su ubicación en Internet. La ruta del archivo que se busca en Internet es `http://arquimedes.matem.unam.mx/Descartes5/lib/`. Esta opción tiene la ventaja de que siempre se leerá el archivo más actual, pero la desventaja de que los archivos Descartes guardados bajo esta opción no funcionarán en ausencia de conexión a Internet.
- Librería portable: A la misma altura del archivo que se guarda, se genera una carpeta */lib* dentro de la cual se hace una copia del archivo *descartes-min.js*.

- Librería de proyecto: Se usa cuando varios archivos creados en Descartes, y que se encuentran al mismo nivel entre ellos, han de compartir el mismo intérprete. Esto ocurre frecuentemente en proyectos, en que una misma unidad consta de varias páginas html de Descartes que van agrupadas. En este caso, la carpeta */lib* que contiene al intérprete *descartes-min.js* se ubica un nivel por arriba de donde se encuentran las escenas agrupadas. Por ejemplo, si los archivos html se guardan en *c:/Proyecto/escenas/*, la ubicación de la carpeta */lib* en este caso será en *c:/Proyecto/*.

En la parte inferior del Editor de Descartes hay una barra de estado. En ella se despliega qué tipo de opción de librería se está usando.

Capítulo 5

El editor de configuraciones

El editor de configuraciones es una ventana nueva en la que propiamente se realiza la programación de un recurso. Consiste en una ventana que se lanza oprimiendo el botón gris pequeño con la letra *E* que se muestra cerca de la esquina superior derecha del Editor de Descartes, como se muestra en la Figura 5.1.

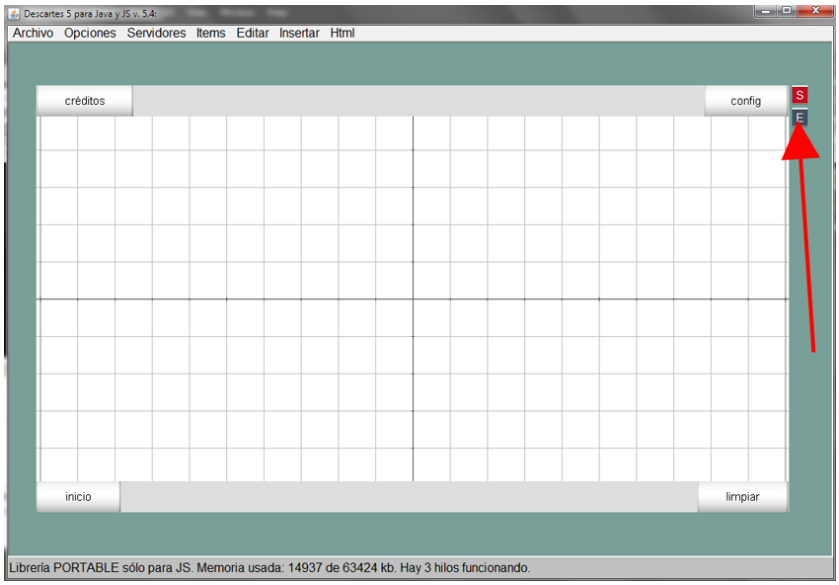

Figura 5.1: Botón para lanzar el editor de configuraciones.

En la figura 5.2 se muestra la ventana del editor de configuraciones. Como se observa, contiene en su parte superior un menú mediante el cual se puede elegir un idioma para mostrar los nombres de los diferentes controles. Por defecto está el español. También incluye un botón *código* con el cual se puede desplegar el código del applet por si fuera necesario copiarlo y pegarlo.

5.1. Selectores

El contenido principal del editor de configuraciones consiste en siete selectores para editar diferentes partes del interactivo. A continuación se da una breve descripción de éstos, aunque posteriormente se abordan a profundidad.

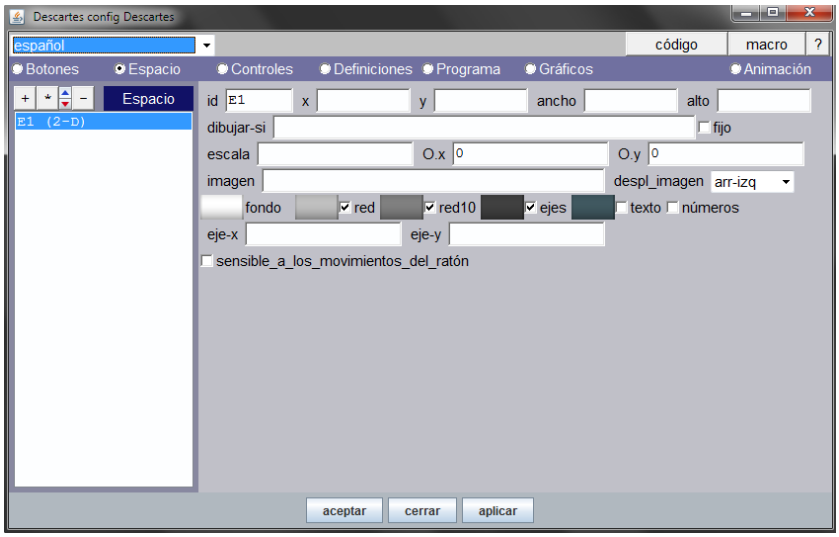

Figura 5.2: Ventana del editor de configuraciones.

El selector *Botones* permite hacer cambios de la interfaz general de la escena con el usuario.

El selector *Espacios* se usa para añadir, duplicar, eliminar o editar las propiedades de los espacios existentes en un interactivo. Los espacios son áreas en las que se muestra material gráfico al usuario.

El selector *Controles* permite añadir, duplicar, eliminar o editar las propiedades de los controles que se encuentran en el interactivo. Dichos controles son aquellos que el usuario puede directamente manipular.

El selector *Definiciones* permite añadir, duplicar, eliminar o editar las propiedades de los elementos que componen la programación como tal del interactivo. Esta parte contiene elementos dentro de los cuales se encuentra el código fuerte dentro de un interactivo.

El selector *Programa* contiene los algoritmos y eventos. Al igual que las definiciones, éstos contienen parte del código que permite el funcionamiento del interactivo. En particular, se refiere a la preparación inicial del interactivo para que esté listo para usarse, pero también a instrucciones que se repiten siempre o dadas ciertas condiciones.

El selector *Gráfico* permite añadir, duplicar, eliminar o editar las propiedades de los gráficos mostrados en los espacios.

El selector *Animación* permite editar las instrucciones y condiciones de una animación en caso de haberla.

Todos estos selectores se abordan a profundidad como capítulos individuales mostrados más adelante.

5.2. Botones del editor de configuraciones

Adicionalmente, el editor de configuraciones cuenta con tres botones en la parte inferior, los cuales se detallan a continuación:

- *aceptar*: Siempre que se hace un cambio en la escena, éste no se muestra de forma inmediata. Al presionar el botón en cuestión, el Editor de configuraciones se cierra y el interactivo se recarga con la última información aplicada.

- *cerrar*: Este botón cierra el Editor de configuraciones pero sin aplicar los cambios. Es decir, cualquier cambio hecho no aparecerá en el interactivo tras cerrar el Editor de configuraciones. Sin embargo, si se vuelve a abrir el Editor, se podrá observar que los cambios están aún presentes, de tal forma que presionar Aceptar finalmente los pasará adelante al interactivo.
- *aplicar*: Este botón aplica los cambios de tal forma que el interactivo se recargará con los últimos cambios hechos. Es decir, hace lo mismo que el botón *acpetar*, con la salvedad de que el editor se mantiene abierto.

Es importante notar que, cuando se manejen las animaciones, muchas veces es más conveniente usar *aceptar* en lugar de *aplicar*, debido a que el inicio automático de algunas animaciones sólo se podrá ver usando dicho botón.

También es importante tener en mente que presionar *aceptar* o *aplicar* no implica que la escena se guarde. Es decir, se pueden hacer cambios en el Editor de configuraciones y, por ejemplo, presionar *aceptar*, con lo que el interactivo se recargará con los cambios hechos. Sin embargo, el archivo hmtl no tendrá estos cambios hasta que se utilice la opción de *Guardar* en el menú *Archivo*.

Capítulo 6

El selector *Botones*

Botones permite controlar las configuraciones más generales.

Como se observa también en la Figura 5.1, hay un botón en cada una de las cuatro esquinas:

- créditos: Despliega en una ventana emergente los créditos y licencia de la herramienta Descartes.
- config: Otro botón que muestra el Editor de configuraciones ya visto anteriormente.
- inicio: Recarga la escena desde el inicio, como si recién hubiera sido abierta, y quitando cualquier modificación que el usuario pudiera haber hecho desde que fue abierta.
- limpiar: Quita rastros gráficos que pudieran haber quedado por interacción del usuario. Los rastros se detallan más adelante.

Estos botones aparecen por defecto en todas las escenas nuevas. Pueden llegar a ser útiles en el proceso de programación, pero no suelen estar presentes en interactivos en su etapa final.

De tal forma que pueden eliminarse, y es precisamente usando el selector de *Botones* que esto se logra con las casillas de verificaciones (de ahora en adelante llamadas *checkbox*) en la parte superior de dicho selector. El eliminar estos botones adicionalmente permite un espacio mayor de trabajo.

Nota: Una función útil en general en todo el Editor de Configuraciones es el despliegue de una ventana emergente de información de cada elemento. Ésta se muestra al hacer clic en el nombre de algún control, o clic derecho en el control mismo. Aunque no todos los controles aún tienen información en dicha ventana, los más comúnmente utilizados sí la tienen. Por ejemplo, hacer clic derecho en el checkbox *Limpiar* muestra su información, como se ve en la Figura 6.1.

El selector *Botones* cuenta con otro control general útil al planear las escenas. Éste es el *signo decimal*. Por defecto, indica que la coma se utilizará como signo decimal (como se hace en muchos países europeos). Sin embargo, puede cambiarse por punto, que permite el uso del punto decimal como se usa en países como México.

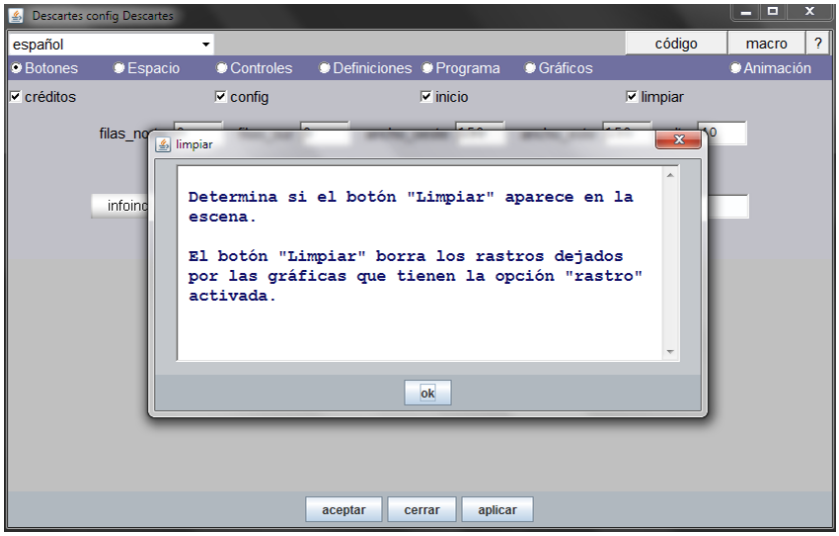

Figura 6.1: Información contextual del checkbox *Limpiar*.

Capítulo 7

El selector *Espacio*

Espacio permite controlar todos los espacios (inicialmente mostrados como planos cartesianos) de la escena. Es sobre estos espacios que se trazan los textos y gráficos mostrados al usuario.

Por defecto, siempre se pinta un espacio bidimensional inicialmente al ejecutar el Editor de Descartes. Este espacio se visualiza en el selector *Espacio* normalmente como *E1* en el panel con la lista de espacios. En la Figura 7.1 se puede observar el selector *Espacio*.

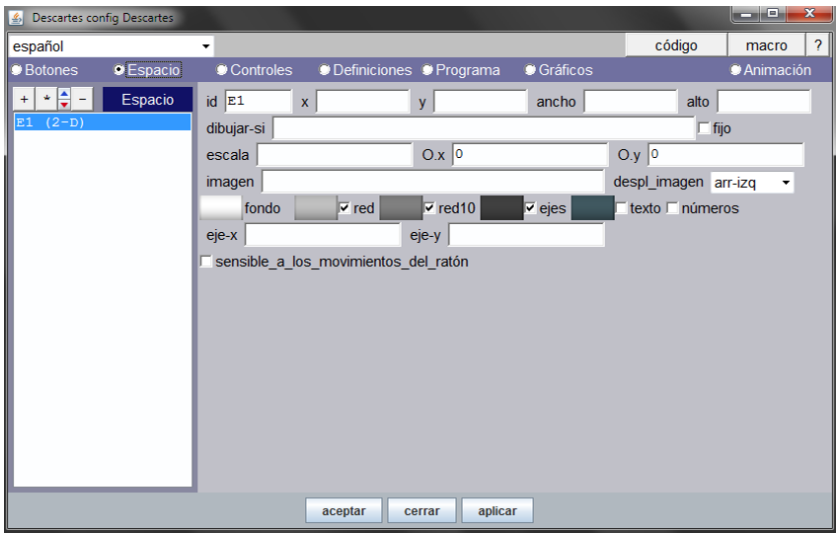

Figura 7.1: Visualización del selector *Espacio*.

El espacio mostrado por defecto tiene un identificador *E1*, que se observa en el campo de texto '*id*'. La mayoría de los otros controles son campos de texto que se encuentran vacíos. Ello se debe a que este es el espacio principal y, si no se determina su posición y tamaño, por defecto abarcará el tamaño de la escena. Este tamaño de la escena por defecto es 970 px (pixeles) por 550 px. En adelante, esto se abreviará como 970 x 550 px. Esta información puede revisarse abriendo el archivo html guardado con cualquier editor de texto.

7.1. Espacio R2 o bidimensional

A continuación se muestra una descripción de los controles del selector *Espacio*:

- **id**: el identificador del espacio. Es el nombre con el que el programa identifica al espacio en cuestión. Los identificadores, de forma general, suelen empezar con letras. El primer caracter de un identificador no debe ser un número.
- **x**: la coordenada horizontal, en pixeles, de la esquina superior izquierda del espacio. Se comienza a medir desde la esquina superior izquierda del Editor para contar. Así pues, la esquina superior izquierda del espacio estará a x pixeles a la derecha de la esquina superior izquierda del espacio de trabajo del Editor.
- **y**: la coordenada vertical, en pixeles, de la esquina superior izquierda del espacio. Se comienza a medir hacia abajo desde el borde superior del espacio de trabajo del Editor. Es decir, el espacio estará a y pixeles por debajo del margen superior del espacio de trabajo del Editor.
- **ancho**: el ancho, en pixeles, del espacio. Si se agrega el sufijo `%`, se tomará el número indicado como el porcentaje de ancho del tamaño de la escena en lugar de considerarlo como el número de pixeles.
- **alto**: el alto, en pixeles, del espacio. Al igual que para el ancho, si se agrega el sufijo `%`, se tomará el número indicado como el porcentaje de altura del tamaño de la escena en lugar de considerarlo como el número de pixeles.
- **dibujar-si**: indica una condición booleana que se evalúa para determinar si el espacio se muestra o no. Muchos distintos elementos de Descartes tienen este control, lo cual lo hace muy versátil para mostrar u ocultar elementos. Por ejemplo, si uno introduce una expresión como $2 == 1$, el programa compara 2 con 1. Como no son iguales, considera que la expresión es falsa y vale 0. Por ello, el espacio no sería trazado. Si, por el contrario, se utiliza $1 == 1$, al ser verdadera la expresión se le otorga un valor de 1. En este caso, el espacio sí sería trazado. Igualmente, se puede introducir directamente 1 ó 0 en este control para trazarlo o no trazarlo, respectivamente. El abordaje de las condiciones booleanas se profundiza más adelante.
- **fijo**: es un checkbox que, de estar marcado, fija el espacio cartesiano de tal manera que el usuario no puede mover ni su posición ni su escala con el mouse o pantalla táctil. Si el espacio no está fijo, el usuario puede mover el espacio cartesiano arrastrándolo y también puede hacer un acercamiento o alejamiento al mismo. Estas funciones se inhabilitan si el checkbox en cuestión está activado. Es importante mencionar que aún con este checkbox marcado, es posible controlar la escala y posición de un espacio cambiando dentro del programa los valores de sus variables intrínsecas de escala y localización. Para más detallas al respecto se puede consultar el apartado sobre variables de espacio.
- **escala**: indica el número de pixeles que componen una unidad en el plano cartesiano. Por defecto (si el campo de texto está vacío) el valor es 48. Es decir, una unidad estará compuesta por 48 pixeles. Disminuir este valor corresponde a hacer un alejamiento, mientras que aumentarlo corresponde a un acercamiento.
- **O.x**: es el desplazamiento en pixeles, sobre la horizontal, del origen del plano cartesiano hacia la derecha. Por ejemplo, un valor de 100 en este campo de texto implica que el plano cartesiano no estará situado en el centro del editor, sino 100 pixeles a la derecha.
- **O.y**: es el desplazamiento en pixeles, sobre la vertical, del origen del plano cartesiano hacia abajo. Por ejemplo, un valor de -100 en este campo de texto implica que el plano cartesiano no estará situado en el centro del editor, sino 100 pixeles hacia arriba del centro.
- **imagen**: corresponde a la ruta relativa de un archivo de imagen que puede presentarse como fondo del espacio. Descartes maneja sin problema imágenes png y jpg. Por ejemplo, si un archivo *portada.png* está en una carpeta */images* que está en el mismo nivel que el archivo de Descartes, entonces se introduciría *images/portada.png* en el campo de texto para desplegar dicha portada como fondo.
- **despl_imagen**: es un menú que determina cómo desplegar una imagen que es de menores dimensiones que el espacio que la contiene.

-arr-izq: la imagen se despliega pegada a la esquina superior izquierda del espacio.

-expand.: la imagen se estira horizontal y verticalmente hasta cubrir el espacio.

-centrada: la imagen se despliega en el centro del espacio.

-mosaico: la imagen se repite en forma de mosaico hasta cubrir todo el espacio.

- **fondo**: es un botón que lanza una ventana emergente para controlar el color del fondo del espacio. El uso de esta ventana se detalla bajo las herramientas generales.
- Controles de color y presentación de ejes, redes y texto en el espacio: consisten un un botón para el color, así como de un checkbox para determinar si se muestran o no los siguientes elementos:

  -red: la red del plano cartesiano.

  -red10: una red de trazos más gruesos que se muestra cada 10 unidades.

  -texto: muestra las coordenadas del punto en que se hace clic en el espacio.

  -números: muestra algunos valores sobre los ejes.
- **eje-x**: muestra un título para las abscisas al lado de su eje.
- **eje-y**: muestra un título para las ordenadas al lado de su eje.
- **sensible a los movimientos del ratón**: es un checkbox que determina si se refrescará la escena y se checarán los eventos cada vez que el mouse se mueve sobre el espacio. Este control no suele usarse ya que puede ralentizar la escena. Además, se vuelve obsoleto si la escena está hecha para visualizarse en un dispositivo móvil. Los eventos a los que se hace referencia se detallarán cuando se traten los eventos bajo el selector *Programa*.

7.2. Espacio R3 o tridimensional

Hasta ahora se ha lidiado con espacios bidimensionales. También existe la posibilidad de usar espacios tridimensionales. Éstos son un tipo de espacio en que se pueden colocar objetos tridimensionales. Como se muestra en la Figura 7.2, una vez que se ha añadido uno de estos espacios y se ha pulsado el botón aplicar, aparece hasta arriba del editor de configuraciones un nuevo selector llamado *gráficos 3D*, en el que se pueden incluir gráficos de tipo tridimensional. Para mayor información se puede consultar el apartado sobre los gráficos 3D.

La mayoría de los controles de estos espacios son iguales a los de espacios bidimensionales, con algunas excepciones, a saber:

- **despliegue**: al usar tres dimensiones, los objetos colocados en el espacio pueden estar unos frente a otros. Para determinar cuáles quedan frente a cuáles hay 3 métodos distintos:

  -orden: pinta los objetos de atrás hacia adelante. Involucra menos cálculos y es el más rápido, pero suele fallar para objetos muy grandes.

  -pintor: dibuja primero los objetos que son tapados por otros. Involucra más cálculos y es más lento, pero es más preciso que el método *orden*.

  -trazado de rayos: rellena los colores pixel a pixel, usando el color del objeto más cercano al observador. Aunque es muy preciso, es muy lento y se recomienda sólo para objetos pequeños.
- **cortar**: es un checkbox que por defecto se encuentra inactivo. Es útil cuando cuando superficies u objetos tridimensionales se cortan, pues permite un correcto despliegue de los objetos. Si los objetos desplegados no se cortan, no es necesario marcar esta opción.

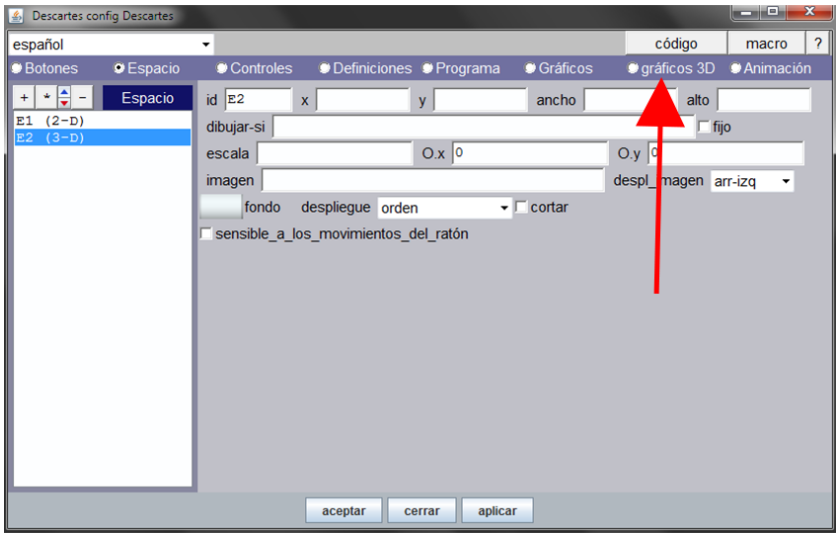

Figura 7.2: Visualización de la localización de los *Gráficos 3D*.

7.3. Espacio HTMLIFrame

Es un espacio en el que se puede mostrar alguna página web con formato html. Este tipo de espacio contiene un campo *archivo* que los demás no tienen. En él se teclea la dirección web o local al html que se desea abrir en dicho espacio. Cabe mencionar que, tras aplicar los cambios, el editor no muestra la página web deseada, sino un bloque donde sólo se ve cuál es la dirección. Pero la página es visible si el interactivo se guarda y se abre en un navegador.

7.4. Panel de espacios en el selector

Adicional a los controles del selector *Espacio*, existe un panel a la izquierda que permite crear nuevos espacios, seleccionar los espacios a editar, duplicar espacios, eliminarlos y alterar el orden de los mismos, como se muestra en la Figura 7.3.

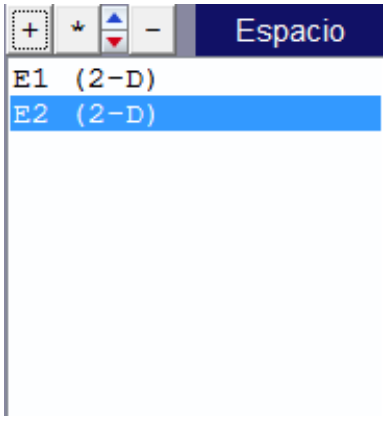

Figura 7.3: Panel izquierdo del selector *Espacio*.

- +: Un botón que lanza una ventana en la cual se puede introducir el nombre de un nuevo espacio y aceptar para crearlo. Los espacios creados de esta forma aparecen con la misma configuración

de un espacio por defecto. Cabe mencionar que se pueden agregar también espacios *3D*. A diferencia de los *2D*, éstos permiten representar objetos tridimensionales. Para añadir un espacio tridimensional, en el diálogo *agregar* hay que seleccionar *R3* en lugar de *R2*, asignar un nombre al espacio y aceptar.

- *: Un botón que lanza una ventana en la cual se puede introducir el nombre de un espacio que, más allá del nombre, será un copia idéntica del espacio seleccionado. Este botón es útil cuando varios espacios han de compartir mucha información.
- pulsador de orden: Dos botones: uno consiste en una flecha hacia arriba y el otro en una flecha hacia abajo. Si se cuenta con una lista de espacios, en ocasiones es necesario alterar el orden en que aparecen. Una vez seleccionado un espacio, que se muestra con un fondo azul como en el ejemplo de la Figura 7.3, presionar la flecha hacia arriba resultará en subirlo un nivel, mientras que presionar la flecha hacia abajo resultará en bajarlo. El orden en que se muestran los espacios en el panel es importante dado que los espacios se pintan en el orden en que aparecen. En el ejemplo de la Figura 7.3, primero se trazará el espacio *E1* y sobre éste se trazará el *E2*.
- -: Un botón que lanza una ventana de confirmación para eliminar el espacio seleccionado.
- : *Espacio*: Un botón azul oscuro con la palabra *Espacio* dentro. Lanza una ventana donde se puede realizar la edición manual de todos los espacios del interactivo. En algunos casos, cuando se desea hacer cambios similares en varios espacios, puede resultar engorroso tener que moverse de espacio en espacio para realizar el mismo cambio. En este caso, la ventanaa mostrada permite realizar los cambios en el código directamente para ahorrar tiempo. La desventaja es que hay que ser muy cuidadoso al usar esta ventana puesto que la eliminación o adición accidental de caracteres especiales puede hacer al interactivo inservible. Por lo mismo, se recomienda no usar esta opción hasta tener cierto grado de conocimiento del funcionamiento de Descartes.

Cabe hacer notar que cuando se selecciona un espacio en la lista del panel a la izquierda del selector, la edición tendrá lugar sobre el espacio seleccionado.

Hagamos un ejercicio para practicar los espacios. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Espacio 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Espacio_01/Espacio_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*. Se recomienda aplicar los cambios tras cada paso y guardar frecuentemente al hacer los ejercicios.

Capítulo 8

El selector *Gráficos*

Nos adelantamos al último de los selectores, el selector *Gráficos* pues los gráficos constituyen el siguiente paso natural después de los espacios. Los gráficos consisten de una variedad de figuras y textos que permiten enriquecer el interactivo.

Muchos de estos gráficos comparten funcionalidad entre ellos. Así pues, esta funcionalidad se describe independientemente de cada gráfico que la contiene al final del presente tema y se omite en la descripción de controles. Es decir, la descripción de controles de un gráfico en particular sólo involucra los controles únicos a ese gráfico. No obstante, cuando se hace referencia a alguna funcionalidad común a todos los gráficos, se incluyen hipervínculos a la explicación de la misma.

Todos los gráficos tienen en común un panel a la izquierda semejante al del selector *Espacio*. En este panel se muestra una lista de los gráficos existentes. Al igual que en el caso del selector *Espacio*, cuenta con un botón para añadir un gráfico nuevo, uno para duplicar el gráfico seleccionado, uno control para mover el gráfico seleccionado arriba o debajo de la lista y un botón para eliminar el gráfico seleccionado. En la Figura 8.1 se observa el selector *Gráficos*.

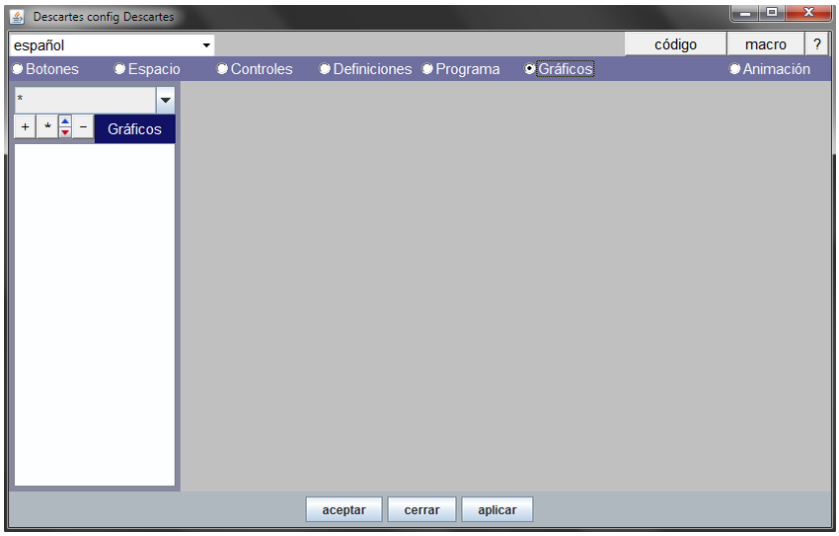

Figura 8.1: Visualización del selector *Gráficos*.

Los gráficos sólo pueden existir dentro de un espacio. De tal forma que existe un menú en la parte superior del panel izquierdo que despliega los espacios existentes. Este menú funciona como un filtro.

Por defecto, muestra un símbolo de asterisco (*), que quiere decir que se muestran todos los gráficos de todos los espacios, que corresponde a cuando el filtro está desactivado. Si se tienen varios espacios y el menú se coloca a alguno de éstos, sólo se mostrarán los gráficos en el espacio seleccionado en el menú, lo que permite una más fácil identificación de los gráficos en caso de haber muchos.

Cuando hay más de un gráfico en un espacio y los gráficos se intersecan, se pintarán en el orden en que aparecen en el panel izquierdo. Es decir, el de hasta arriba será el primero en pintarse y el de hasta abajo, que será el último en pintarse, cubriría a los que se encuentran arriba de él en la lista.

También al igual que en el selector *Espacio* hay un botón con el texto *Gráficos* que lanza una ventana con el código de los gráficos en forma de texto. Es importante que, aunque el filtro de algún espacio en particular esté activo, este botón mostrará el texto de todos los gráficos de todos los espacios.

8.1. Gráficos 2D

Éstos constituyen los gráficos más comúnmente usados, que son bidimensionales. Se pintan sólo en espacios de dos dimensiones.

8.1.1. Gráfico *Ecuación*

Este gráfico consiste en una curva correspondiente a una ecuación. En la figura 8.2 se muestra un gráfico tipo *Ecuación*, mientras que en la figura 8.3 se observa cómo deben ajustarse los controles del gráfico para que quede así trazado.

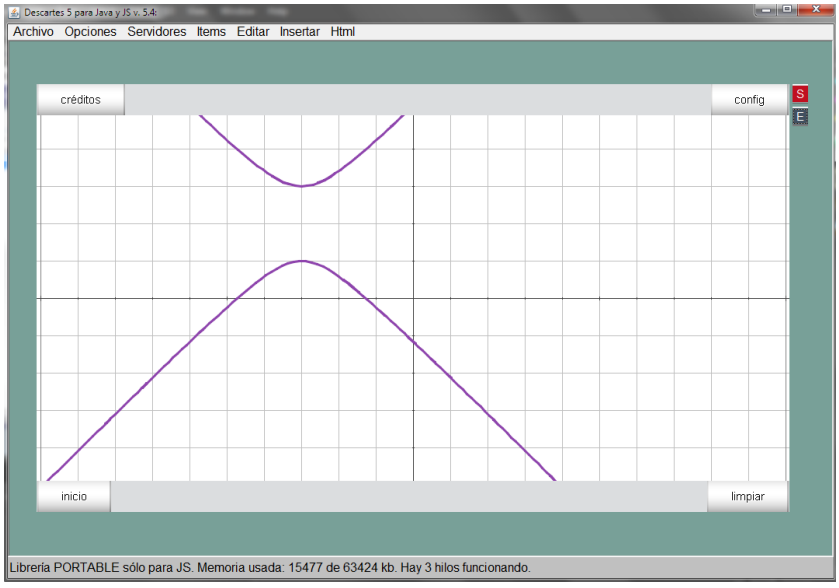

Figura 8.2: Ejemplo del gráfico *Ecuación*.

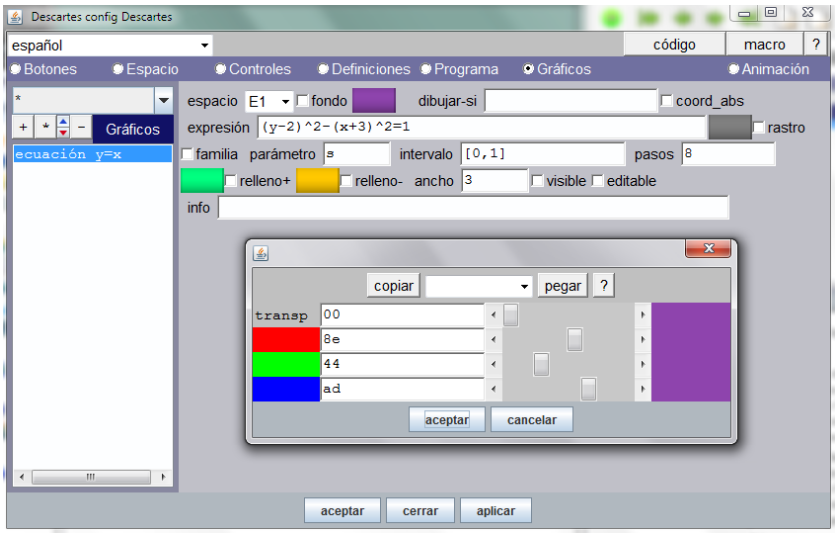

Figura 8.3: Configuración del ejemplo del gráfico *Ecuación*.

- **expresión**: es un campo de texto donde se introduce la ecuación que desea graficarse. Ocupa las variables y y x para referirse a la ordenadas y abscisas respectivamente en el plano. Es realmente una ecuación la que se puede trazar. Por ejemplo, $y = x$ es la que aparece por defecto, pero bien podría graficarse $x^2 + \frac{y^2}{2} = 1$ para una elipse, o incluso $x = 3$. Es decir, la ecuación no tiene que forzosamente ser una función de y en x.
- **relleno+**: es un checkbox que, de estar marcado, permite que se rellene el área entre el eje y la gráfica con el color determinado por el control de color al lado del checkbox. Por ejemplo, para la expresión $y = x^3$, se rellenará la parte entre el eje x y la gráfica que queda por encima del eje con el color seleccionado. Si la gráfica es de la forma $x = y^3$, se rellenará el área entre el eje y y la gráfica a la derecha de este eje.
- **relleno-**: es un checkbox que, de estar marcado, permite que se rellene el área entre el eje y la gráfica con el color determinado por el control de color al lado del checkbox. Por ejemplo, para la expresión $y = x^3$, se rellenará la parte entre el eje x y la gráfica que queda por debajo del eje con el color seleccionado. Si la gráfica es de la forma $x = y^3$, se rellenará el área entre el eje y y la gráfica a la izquierda de este eje.
- **visible**: es un checkbox que permite visualizar en el interactivo la ecuación que está graficada si se encuentra marcado. La ecuación se visualiza en una barra en la parte inferior del interactivo.
- **editable**: es un checkbox que funciona sólo si el control *visible* está marcado. Permite, además de ver la ecuación en el interactivo, que ésta sea modificada, resultando en que el gráfico cambie dinámicamente.
- Los controles **fondo**, **dibujar-si**, **coordenadas absolutas**, **rastro**, **familia**, **parámetro**, **intervalo**, **pasos**, **ancho** e **info** se discuten al final del presente tema. Los botones de controles del color del gráfico, y el de los controles *relleno+* y *relleno-* son los genéricos explicados en la herramienta de control de colores.

8.1.2. Gráfico *Curva*

Este gráfico consiste en varios trazos que unen puntos determinados por un parámetro. El parámetro es una variable que adopta valores en un intervalo en un número dado de pasos. De tal forma que se puede ver como una serie de segmentos uno tras otro unidos por sus extremos. En la Figura 8.4 se muesta un ejemplo de una curva, y en la Figura 8.5 se muestra cómo se configura dicho ejemplo.

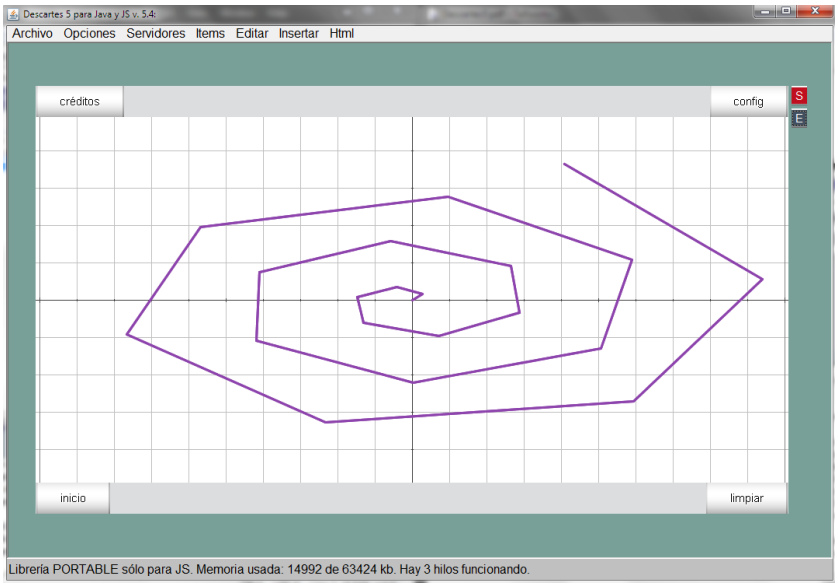

Figura 8.4: Ejemplo del gráfico *Curva*.

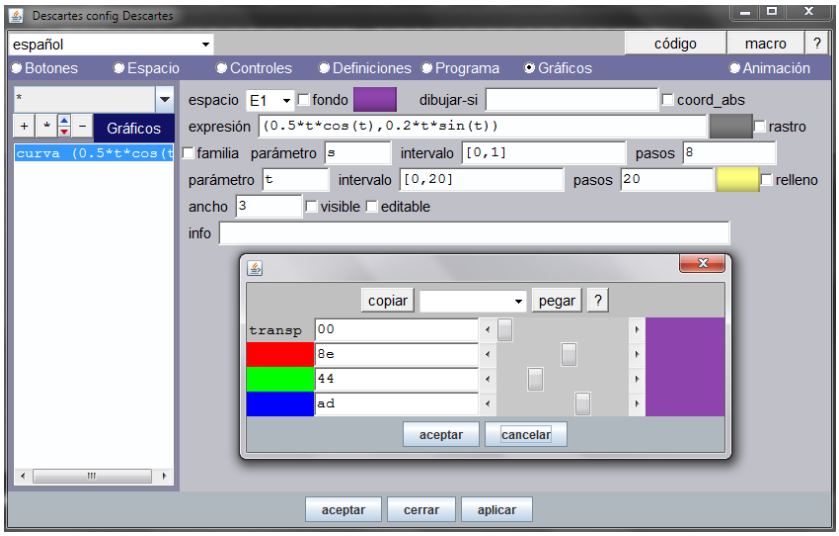

Figura 8.5: Configuración del ejemplo del gráfico *Curva*.

- **expresión**: es un campo de texto donde se escribe la sucesión de puntos que representan el par ordenado de los extremos de los segmentos que componen la curva. Para el gráfico *curva*, la expresión es del tipo (t, t). Esto es, es un par ordenado de funciones de la variable t, que es el parámetro del que depende la curva, y que se detalla abajo.
- **parámetro**: es un campo de texto donde se introduce el nombre de la variable de la cual depende la curva. La variable por defecto es t. Es importante hacer la diferencia entre el parámetro de una familia (cuya variable por defecto es s) y el parámetro de la curva, puesto que puede prestarse a confusión.
- **visible**: es un checkbox que permite visualizar en el interactivo la expresión de la curva graficada si se encuentra marcado. Esta expresión se visualiza en una barra en la parte inferior del

interactivo.

- **editable**: es un checkbox que funciona sólo si el control *visible* está marcado. Permite, además de ver la expresión de la curva en el interactivo, que ésta sea modificada, resultando en que el gráfico cambie dinámicamente.

Ahora que conocemos ya dos gráficos distintos, aprovechemos para hacer un ejercicio breve. Se pueden aprovechar los gráficos vistos para observar cómo un polígono de muchos lados puede aproximar una circunferencia.

El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Gráficos 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_01/Graficos_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

8.1.3. Gráfico *Sucesión*

Este gráfico consiste en una serie de puntos que representan una sucesión matemática. Una sucesión matemática está compuesta por puntos en los números naturales en las abscisas y enteros en las ordenadas. Es similar a una función, pero que sólo es válida en donde las abscisas tienen valores en los números naturales $(1, 2, 3, 4, ...)$. Las coordenadas de los puntos de la sucesión se manejan como pares ordenados. En la Figura 8.6 se muestra un ejemplo de este gráfico y en la figura 8.7 se muestra cómo debe configurarse dicho ejemplo.

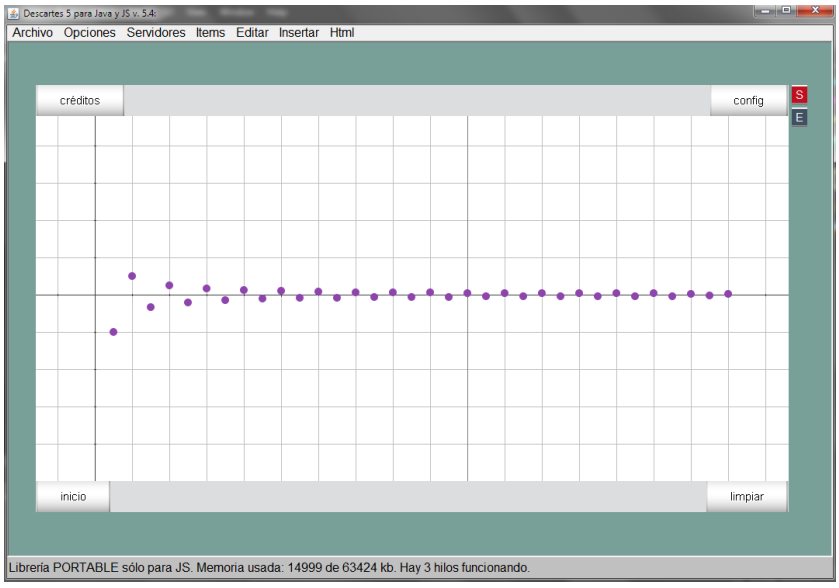

Figura 8.6: Ejemplo del gráfico *Sucesión*.

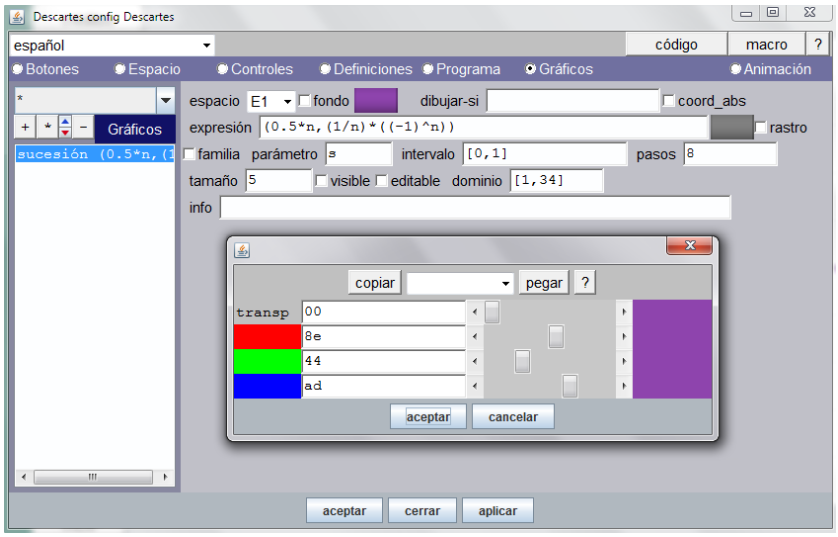

Figura 8.7: Configuración del ejemplo del gráfico *Sucesión*.

- **expresión**: es un campo de texto donde se introduce la expresión de la sucesión como un par ordenado. Por defecto trae la sucesión $(n, 1/n)$. Es decir, el primer punto será el $(1, 1/1)$ o $(1, 1)$; el segundo punto será el $(1, 1/2)$ o $(1, 0.5)$; y así sucesivamente.
- **visible**: es un checkbox que cuando está marcado permite visualizar la regla de correspondencia de la sucesión dentro del interactivo como una barra en la parte inferior de éste.
- **editable**: es un checkbox que cuando está marcado permite que el usuario modifique el texto visible de la regla de correspondencia de la sucesión directamente en el interactivo, permitiendo así que los puntos que representan la sucesión cambien de forma dinámica.
- **dominio**: es un campo de texto donde se introduce el dominio de la sucesión. Por defecto tiene el texto $[1, 100]$, que quiere decir que sólo incluirá los puntos para los cuales la n vale 1 y hasta cuando vale 100.

Hagamos un breve ejercicio para ver cómo funcionan las sucesiones. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Gráficos 02.. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_02/Graficos_02_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

8.1.4. Gráfico *Punto*

Este gráfico consiste en un único punto cuyas coordenadas se dan de forma explícita. En la Figura 8.8 se muestra un ejemplo de este gráfico. En la Figura 8.9 se muestra la configuración para lograr dicho ejemplo.

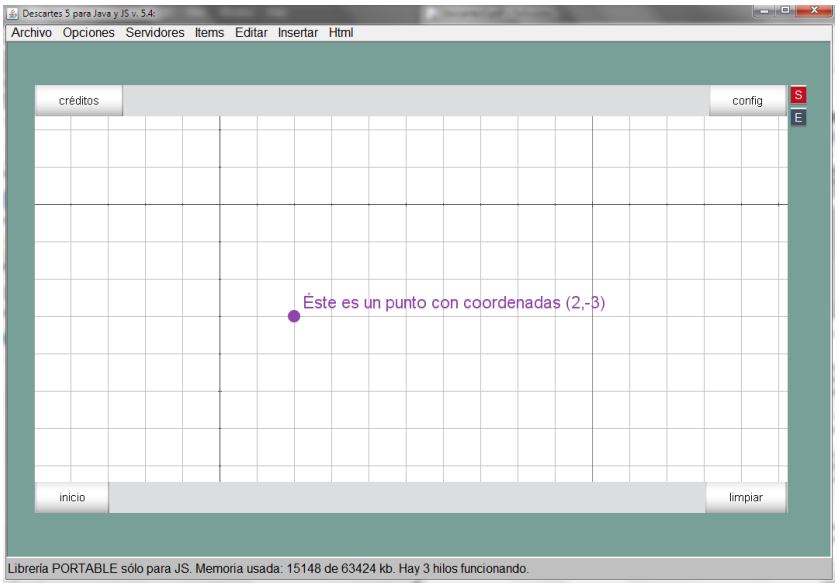

Figura 8.8: Ejemplo del gráfico *Punto*.

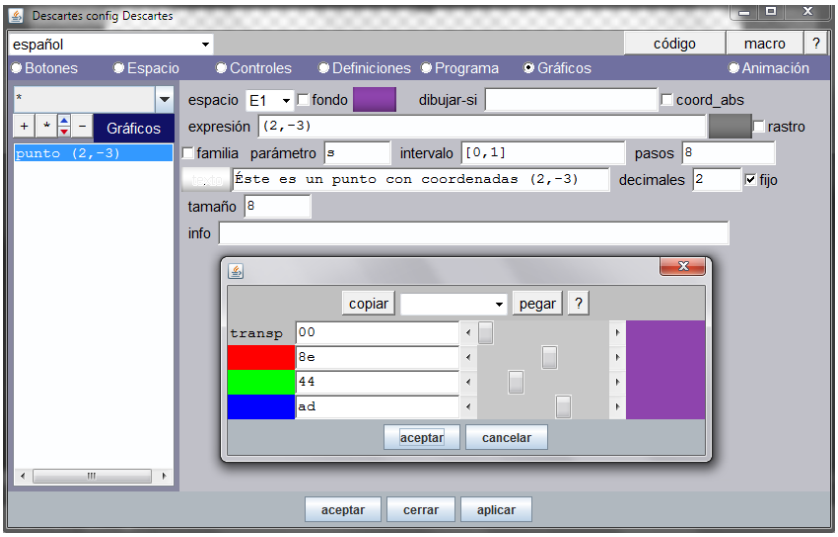

Figura 8.9: Configuración del ejemplo del gráfico *Punto*.

- **expresión**: es un campo de texto en donde se colocan las coordenadas del único punto.
- **texto**: es un campo de texto con un botón mediante los cuales se puede introducir un texto que acompañe al punto. Se puede introducir el texto directamente en el campo de texto, o bien puede desplegarse una ventana para una introducción más cómoda del mismo haciendo pulsando el botón *texto*. Al pulsarse este botón se da la opción de introducir el texto como plano o enriquecido. En el capítulo Herramientas generales se proporciona más información sobre la herramienta de introducción de textos, que está disponible para otros gráficos también.
- **decimales**: es un campo de texto donde se introduce un número entero correspondiente al número de decimales que habrán de mostrarse en caso que el texto contenga alguna variable cuyo valor se quiera visualizar.

- **fijo**: es un checkbox que determina si siempre se muestran el número de decimales elegido en el control *decimales*, aún cuando no sean significativos. Siempre se mostrarán los decimales elegidos si este checkbox está marcado. De lo contrario, sólo se mostrarán si son significativos.

Hagamos un breve ejercicio para ver el funcionamiento de los puntos. Aprovecharemos esta oportunidad para revisar las familias de gráficos (en este caso, sólo del punto en cuestión), un poco sobre los textos y algo sobre coordenadas relativas y absolutas. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Gráficos 03. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_03/Graficos_03_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

8.1.5. Gráfico *Segmento*

Este gráfico consiste en un segmento determinado por las coordenadas de sus extremos. En la Figura 8.10 se muestra un ejemplo de este gráfico. En la Figura 8.11 se muestra la configuración necesaria para lograr este ejemplo.

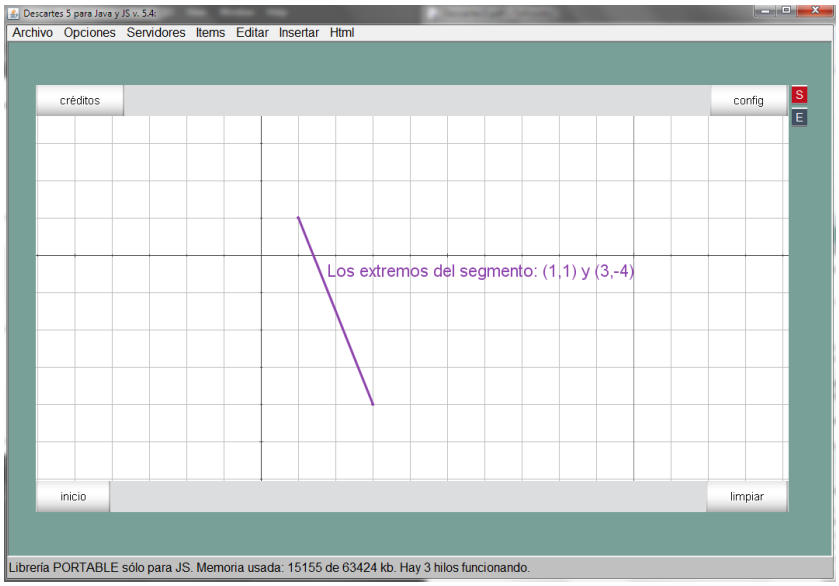

Figura 8.10: Ejemplo del gráfico *Segmento*.

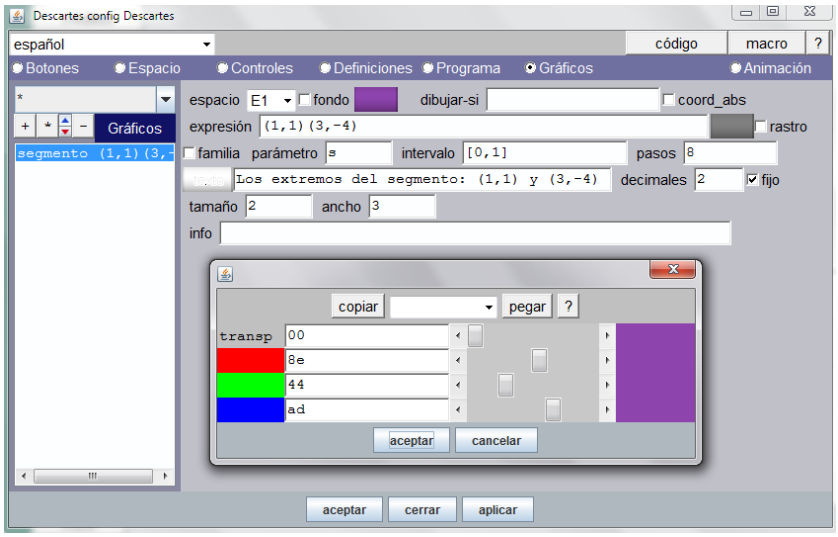

Figura 8.11: Ejemplo del gráfico *Segmento*.

- **expresión**: es un campo de texto en el cual se introducen un par de coordenadas correspondientes a los extremos del segmento.
- **texto**: es un campo de texto, acompañado de un botón para abrir la ventana de edición de texto, mediante el cual se puede introducir un texto que se muestra cercano al punto medio del segmento. La funcionalidad del texto es la misma que la detallada en el apartado de la herramienta de edición de textos.
- **tamaño**: es un campo de texto donde se introduce un número entero correspondiente al tamaño en pixeles de los puntos que flanquean al segmento. Se puede usar el valor 0 si se desea sólo mostrar el segmento y no sus puntos extremos.
- **ancho**: es un campo de texto donde se introduce un número entero correspondiente al ancho en pixeles del segmento mismo.

Hagamos un breve ejercicio para practicar el uso de segmentos. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Gráficos 04. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_04/Graficos_04_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En el ejercicio anterior se pudo notar algo interesante. Los gráficos en coordenadas absolutas, dado que se encuentran definidos con respecto a la esquina superior izquierda del interactivo, no son susceptibles a movimientos del plano cartesiano del espacio, mientras que los que están definidos en coordenadas relativas sí. Esto representa una ventaja de usar coordenadas absolutas para aquellos que se desee permanezcan estáticos.

8.1.6. Gráfico *Flecha*

Este gráfico es casi idéntico al *segmento*, con la salvedad de que en la última coordenada indicada en su campo de texto *expresión* se pinta una punta de flecha. En la Figura 8.12 se muestran un ejemplo de este gráfico. En la Figura 8.13 se muestra la forma de configurar este ejemplo.

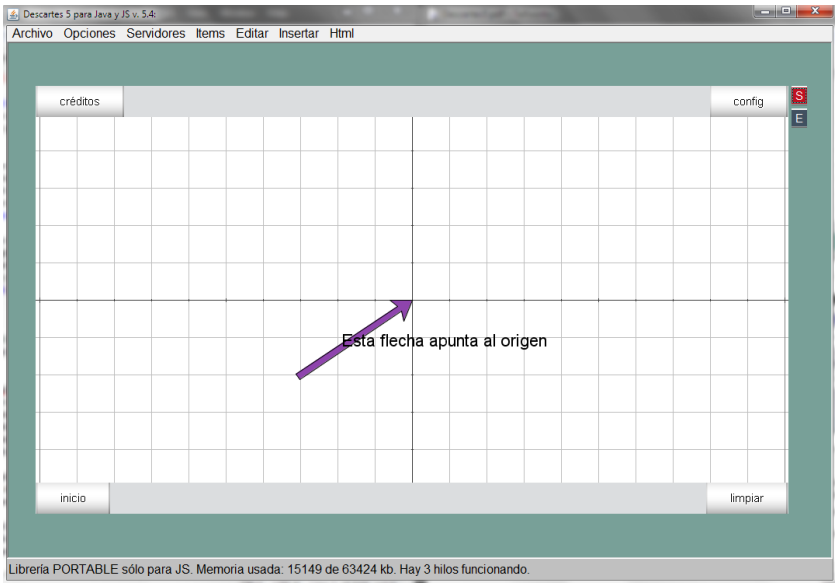

Figura 8.12: Ejemplo del gráfico *Flecha*.

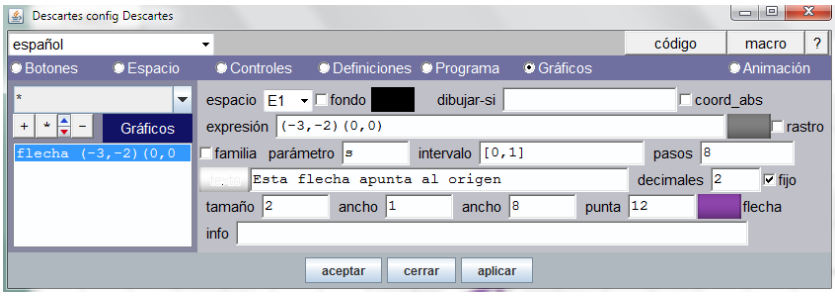

Figura 8.13: Configuración del ejemplo del gráfico *Flecha*.

- **expresión**: es un campo de texto en donde se indica el par de coordenadas de los extremos de la flecha. Como se mencionó anteriormente, la última coordenada corresponde al extremo donde está la punta de la flecha.
- **texto**: es un campo de texto y el botón correspondiente para mostrar las ventanas de introducción de texto. En dicho campo se puede incluir texto que se mostrará cerca del punto medio de la flecha.
- **ancho**: es un campo de texto que determina el grosor en pixeles del ancho de la flecha. En versiones actuales existen dos parámetros *ancho* para la flecha. Sólo el segundo es funcional.
- **punta**: es un campo de texto que determina el tamaño de la punta de la flecha en pixeles.

Debido a la similitud entre el segmento y la flecha, no se hará un ejercicio particular para la flecha.

8.1.7. Gráfico *Polígono*

Este gráfico es trazado a partir de coordenadas de puntos individuales que representan los vértices de un polígono. Las coordenadas de los vértices en secuencia determinan los segmentos que formarán los lados del polígono. En la Figura 8.14 se muestra un ejemplo de este tipo de gráfico. En la Figura 8.15 se muestran la forma de configurar dicho gráfico.

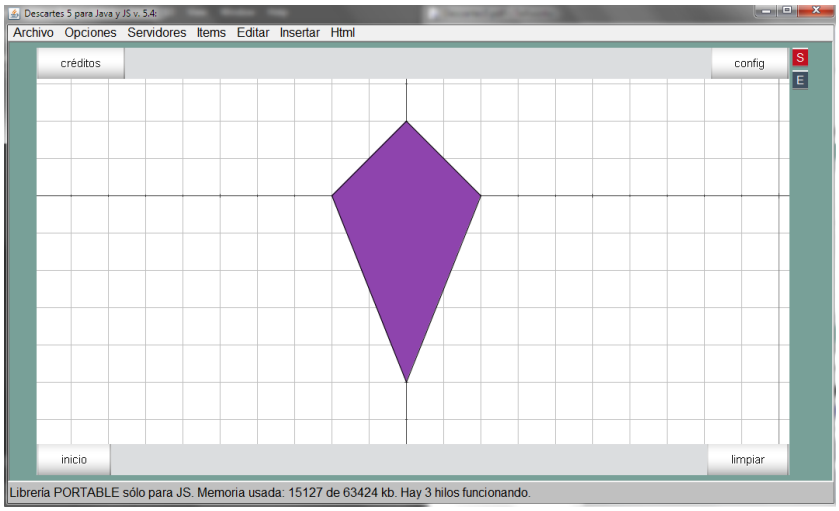

Figura 8.14: Ejemplo del gráfico *Polígono*.

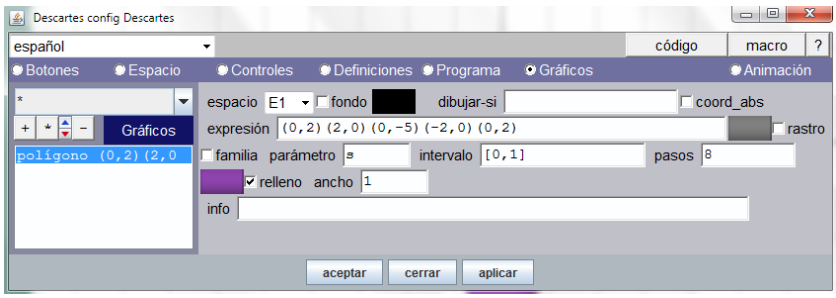

Figura 8.15: Configuración del ejemplo del gráfico *Polígono*.

- **expresión**: es un campo de texto en el cual se introducen las coordenadas de los vértices que componen el polígono. Los lados del mismo se trazarán del primer vértices introducido al segundo, del segundo al tercero y así sucesivamente. A pesar del nombre *polígono*, bien puede consistir en una figura abierta si el primer vértice y el último no coinciden.
- **relleno**: es un checkbox que de estar marcado hará que el interior del polígono tenga un relleno con color. El color se determina a partir del botón de selección de color al lado izquierdo del checkbox, cuya funcionalidad se detalla en la herramienta de control de colores.

Aprovechemos para construir un espacio en el que se podría incluir un texto explicativo en el interior (algo así como una ventana emergente). Es conveniente que dicha ventana tenga un marco, y el marco será el polígono mismo. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Gráficos 05. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_05/Graficos_05_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio tuvimos la oportunidad de ver cómo algunas variables internas de Descartes (en este caso el ancho y alto de un espacio) pueden ser útiles. Una descripción más detallada de estas variables se encuentra en el apartado sobre variables de espacio. Observamos nuevamente la utilidad de las coordenadas absolutas respecto a las relativas.

8.1.8. Gráfico *Arco*

Este gráfico consiste en una parte de una circunferencia. Aunque se puede hacer lo mismo con un gráfico tipo *ecuación*, este nuevo gráfico permite una introducción más cómoda en que se proporciona simplemente el centro, el radio y el radio que habrá de barrer el arco. En la Figura 8.16 se muestra un ejemplo con este gráfico. En la Figura 8.17 se muestra la configuración para obtener dicho ejemplo.

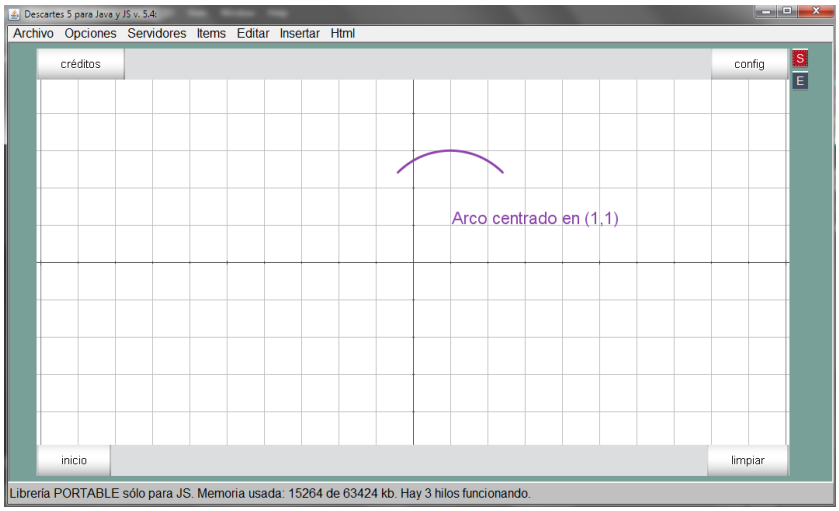

Figura 8.16: Ejemplo del gráfico *Arco*.

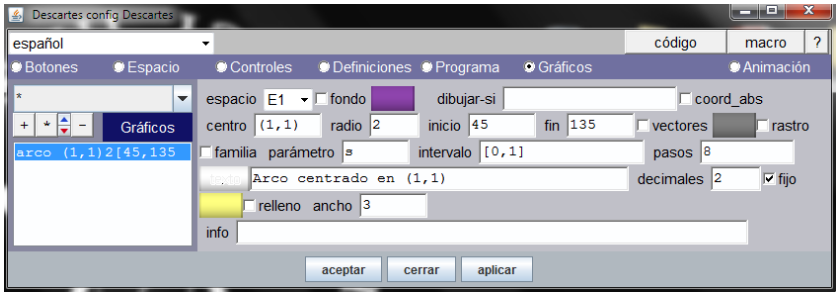

Figura 8.17: Configuración del ejemplo del gráfico *Arco*.

- **centro**: es un campo de texto donde se indican las coordenadas del centro del arco. Por ejemplo $(3, 2)$ implica que el centro del arco estará 3 unidades desplazado a la derecha del origen y 2 unidades hacia arriba.
- **radio**: es un campo de texto donde se indica el radio que habrá de tener el arco.
- **inicio**: es un campo de texto donde se introduce el ángulo inicial en grados a partir del cual se empieza a trazar el arco.
- **fin**: es un campo de texto donde se introduce el ángulo final en grados hasta donde se trazará el arco.
- **vectores**: es un checkbox que cambia la forma de introducir el *inicio* y *fin*. Si no está marcado, el inicio y fin representan los grados de incio y fin del arco. Si se encuentra marcado, *inicio* y *fin* reciben una coordenada de un vector que representa un lado de inicio del arco y el otro el lado del fin. Por ejemplo, se puede introducir $(3, 2)$ como *inicio*. Estos vectores siguen teniendo como origen el centro del arco.

- **texto**: es un campo de texto para introducir texto que ha de mostrarse cerca del centro del arco. Incluye su botón *texto* mediante el cual se puede introducir texto en una ventana de forma más cómoda, y hasta texto enriquecido. El funcionamiento se detalla en la herramienta de introducción de texto. Adicionalmente tiene su campo de texto *decimales* y su checkbox *fijo* para indicar el número de decimales a mostrar de ser necesario, y si éstos han de ser mostrados de forma fija o sólo los decimales significativos.
- **relleno**: es un checkbox que determina si el arco habrá de estar rellenado. Incluye un control de color para elegir el color a usar. El funcionamiento de este control de color se detalla en la herramienta de control de colores.

Hagamos un ejercicio para practicar la funcionalidad de este tipo de gráfico. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Graficos 06. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_06/Graficos_06_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Se pudo observar en este interactivo que los arcos son una forma más inmediata de trazar una parte de una circunferencia cuando se cuenta con el centro y el radio de la misma. En ocasiones, puede resultar útil considerar a esta parte de la circunferencia como flanqueada por vectores. De ahí que sea posible introducir un arco mediante la modalidad *vectores*. Igualmente, si se cuenta con el centro y radio de una circunferencia, es más fácil introducirla como un arco de 360°que mediante una ecuación.

8.1.9. Gráfico *Relleno*

Este gráfico consiste en un relleno de un color seleccionado para el usuario. Los límites de dicho relleno se determinan por los bordes de otros gráficos.

Aunque varios gráficos cuentan con su propio relleno, a veces resulta útil colorear porciones del espacio que no están delimitadas por un solo gráfico, sino por una multitud de ellos. Es en estos casos que conviene usar el gráfico *relleno*. En la Figura 8.18 se muestra un ejemplo de este tipo de gráfico. En la Figura 8.19 se muestra la forma de configurar dicho ejemplo. Note que la figura que se habrá de rellenar es un gráfico tipo ecuación de una hipérbola: $y^2 - x^2 = 1$. El punto de relleno es el origen (0,0), por lo que se rellena el área alrededor de dicho punto.

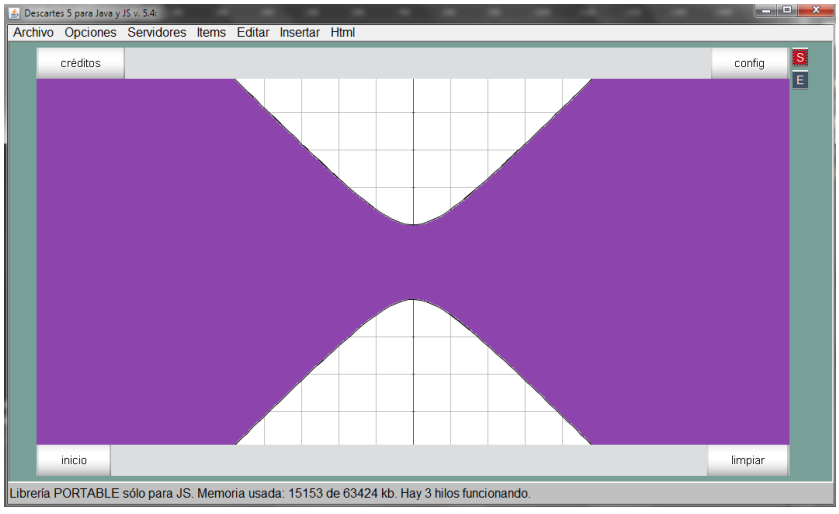

Figura 8.18: Ejemplo del gráfico *Relleno*.

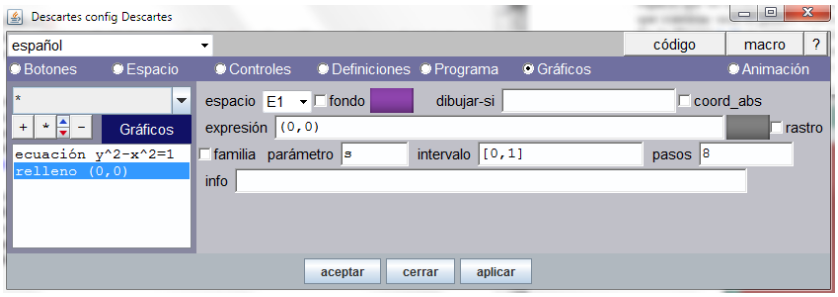

Figura 8.19: Configuración del ejemplo del gráfico *Relleno*.

- **expresión**: es un campo de texto en el que se introduce la coordenada de un punto que habrá de estar dentro del área delimitada que se quiere rellenar con color.

A continuación haremos un ejercicio que involucra varias figuras geométricas que se intersecan entre sí, y cómo el gráfico *relleno* nos puede ayudar a colorear el área delimitada entre dichas figuras. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Graficos 07. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_07/Graficos_07_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Se observó en este ejercicio la utilidad del gráfico tipo *relleno*, que puede usarse, por ejemplo, en ejercicios de cálculo de áreas compuestas por varias figuras. Es importante notar que el relleno busca fronteras que lo limiten. Si no encuentra una, continuará pintando. Por lo mismo, es preciso asegurarse que las áreas estén delimitadas correctamente.

8.1.10. Gráfico *Texto*

Este gráfico consiste en texto que se puede introducir con el fin de explicar algo particular de una lección. Por ejemplo, con este texto se pueden incluir instrucciones, explicaciones de fenómenos, preguntas, etc.

Este gráfico no cuenta con coordenadas relativas. Solamente maneja coordenadas absolutas. Gran parte del funcionamiento está detallado en el apartado sobre la herramienta de introducción de textos. En la Figura 8.20 se muestra un ejemplo de un texto con una fracción continua. En la Figura 8.21 se muestran cómo es la configuración para dicho ejemplo.

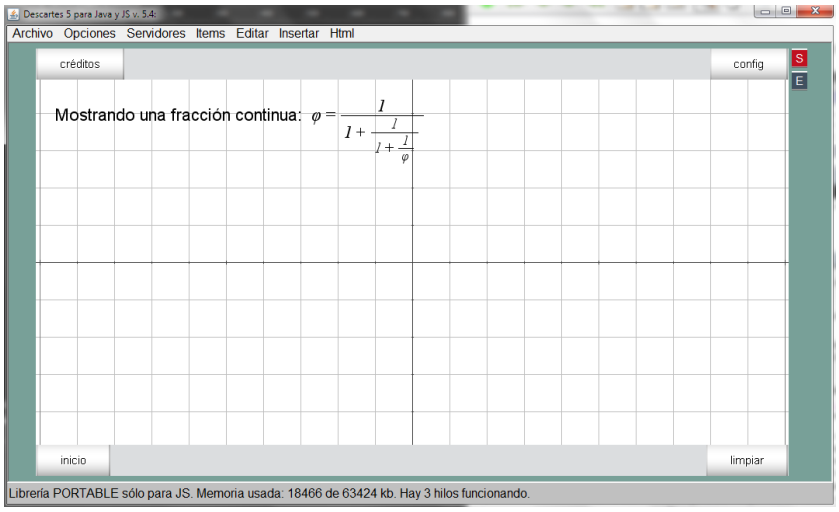

Figura 8.20: Ejemplo del gráfico *Texto*.

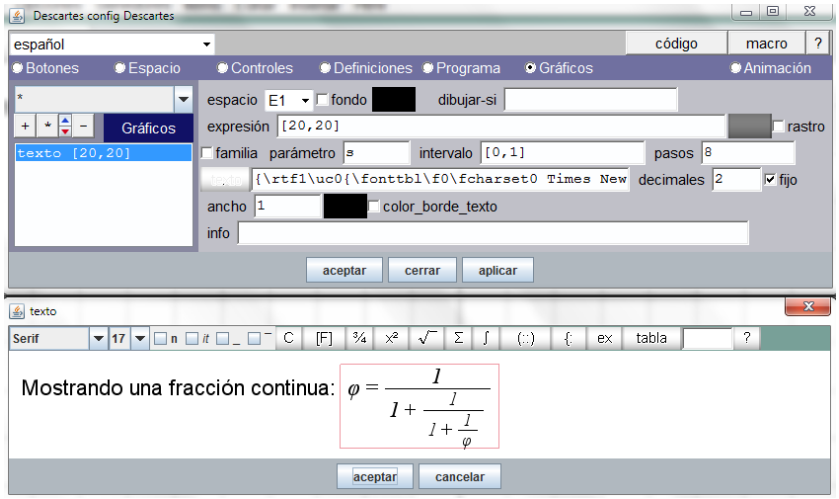

Figura 8.21: Configuración del ejemplo del gráfico *Texto*.

- **expresión**: es un campo de texto donde se introduce la coordenada de inicio del texto (es decir, la esquina superior izquierda donde comienza el texto) entre corchetes [y]. Deben usarse coordenadas absolutas. Si se quiere que la coordenada sea el centro del bloque de texto, puede agregarse $0, 0$ como dos últimas coordenadas. Por ejemplo, si la expresión es [500,100,0,0], entonces el texto quedará centrado en la coordenada absoluta $(500, 100)$ horizontalmente. Adicionalmente, es posible introducir valores distintos de cero para las dos últimas entradas de la expresión. Por ejemplo, $[500, 100, 50, 30]$. En este caso, se indica que se ha de centrar alrededor del punto $(500, 100)$ desplazado de forma centrada 50 pixeles a la derecha y 30 hacia abajo. En realidad, el centro del texto se desplazará a la derecha $\frac{50}{2} = 25$ pixeles (para que el desplazamiento sea centrado se debe dejar lo mismo de margen izquierdo que de derecho) y $\frac{30}{2} = 15$ pixeles hacia abajo.
- **ancho**: es un campo de texto donde se introduce el ancho en pixeles del texto. Cuando se usa texto simple, es posible tener líneas muy largas. En este caso el ancho determinará hasta qué límite se permitirá que aparezca el texto antes de introducir una nueva línea. Esta funcionalidad sólo es válida para texto simple. En caso de introducir texto enriquecido, el cambio de línea

deberá hacerse de forma manual en la ventana de introducción de texto, y el parámetro *ancho* deberá tener un valor de 1.

- **color_borde_texto**: es un checkbox que traza un borde al texto del color elegido en el control de color a su izquierda. Para más detalles sobre el control de color se puede consultar el apartado de la herramienta del control de colores.

A continuación haremos un ejercicio que involucra el gráfico tipo *texto* para aclarar posible dudas. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Graficos 08. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_08/Graficos_08_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

8.1.11. Gráfico *Imagen*

Este gráfico consiste en una imagen tipo jpg, png o gif que puede insertarse en el interactivo. La imagen debe estar en la misma carpeta que el interactivo o en una subcarpeta. En la Figura 8.22 se muestra un ejemplo de un gráfico tipo imagen. En la Figura 8.23 se muestra la configuración necesaria para lograr dicho ejemplo. Note que la imagen usada en este ejemplo se guarda en una carpeta *images* a la altura del archivo del interactivo, y su nombre es *1.png*.

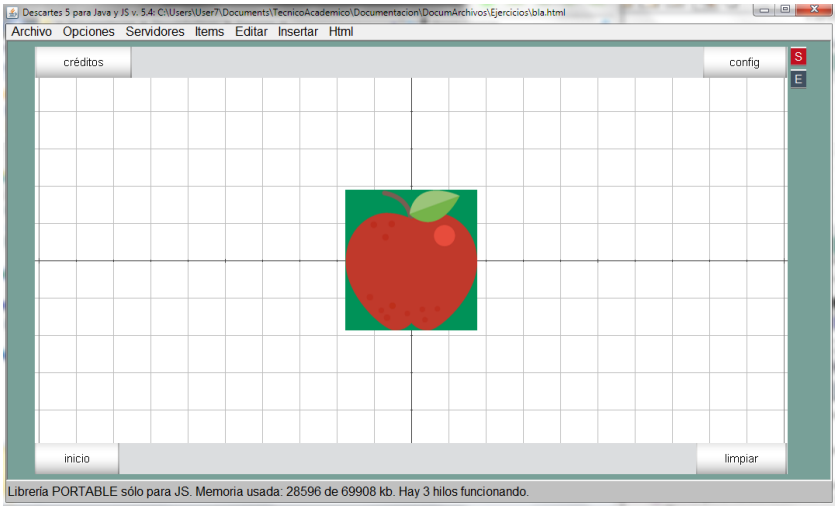

Figura 8.22: Ejemplo del gráfico *Imagen*.

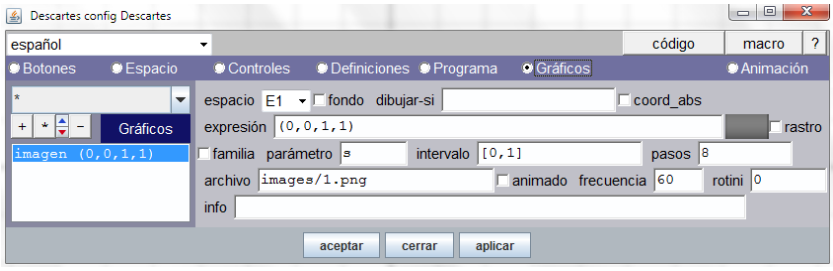

Figura 8.23: Configuración del ejemplo del gráfico *Imagen*.

- **expresión**: es un campo de texto en el que se introducen las coordenadas donde habrá de mostrarse la imagen en el interactivo. Por defecto son dos coordenadas y corresponden a dónde estará situada la esquina superior izquierda de la imagen. No obstante, se pueden introducir cuatro entradas en los paréntesis, en donde la tercera y cuarta entradas corresponden al factor de escala del ancho y alto de la imagen. En caso de definir las cuatro entradas, las dos primeras ya no marcan la esquina superior izquierda de la imagen, sino su centro.
- **archivo**: es un campo donde se indica la ruta al archivo de imagen. En caso de incluirse en una subcarpeta, se usa la diagonal normal (/) para usar las subcarpetas.
- **animado**: es un checkbox que cuando está marcado indica que la imagen es de tipo animado (gif).
- **frecuencia**: es un campo de texto donde se introduce el número de milisegundos que han de transcurrir entre una actualización de la imagen animada y la siguiente. Sólo se usa cuando si la imagen es animada.
- **rotini**: es un campo de texto donde se introduce el ángulo en grados en que aparece rotada la imagen.

Hagamos un ejercicio para practicar el uso de este tipo de gráfico. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Graficos 09. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos_09/Graficos_09_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio se observó que es posible introducir imágenes en los interactivos. A estas se les puede controlar su tamaño y proporción, su colocación y su rotación. Recuerde que aunque todos los parámetros en el ejercicio son números, también pueden usarse variables que permiten hacer estos cambios de forma interactiva. No obstante, esto se verá cuando se aborden ejercicios más avanzados.

Es importante tener en cuenta que es necesario guardar el archivo html antes de que éstas se visualicen. Ello se deba a que es sólo hasta que se ha guardado el archivo que Descartes puede determinar a partir de qué ubicación abordar la ruta relativa donde se encuentra la imagen. Así pues, si sólo se aplican los cambios sin haber antes guardado la escena, ésta no será mostrada.

8.2. Gráficos 3D

Estos gráficos son de tipo tridimensional. Por lo mismo, sólo se pintan en espacios tridimensionales o 3D.

8.2.1. Gráfico *Segmento*

Es igual al gráfico segmento en dos dimensiones, con la salvedad de que las coordenadas de inicio y fin del segmento en el campo *expresión* de éste conllevan tres entradas (una por cada eje en tres dimensiones).

En la Figura 8.24 se muestra un ejemplo del gráfico tridimensional *segmento*. En la Figura 8.25 se muestra la configuración para lograr este ejemplo. Sólo se ve el primero de los segmentos en la configuración. No obstante, note que hay 2 segmentos más. Cada uno tiene longitud 2, y están orientados respectivamente a lo largo de los ejes coordenados. Esto facilita ubicar dónde se encuentran los gráficos tridimensionales en el espacio 3D. De tal forma que este ejemplo se usará subsecuentemente en los siguientes gráficos.

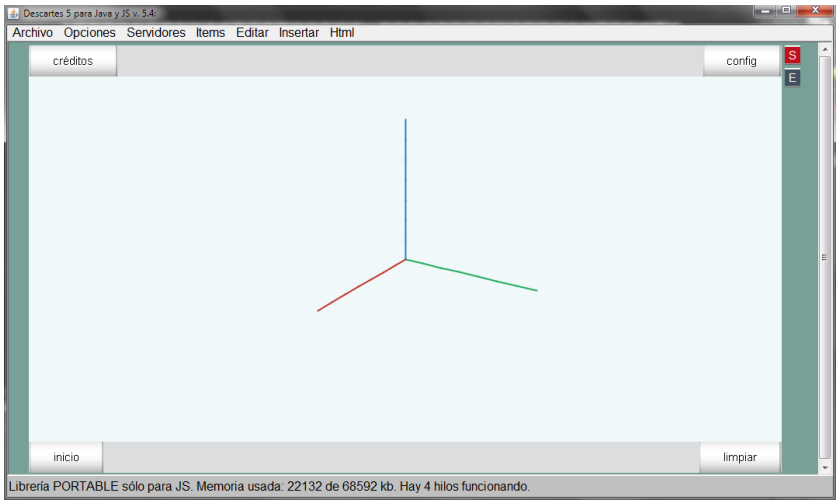

Figura 8.24: Ejemplo del gráfico tridimensional *Segmento*.

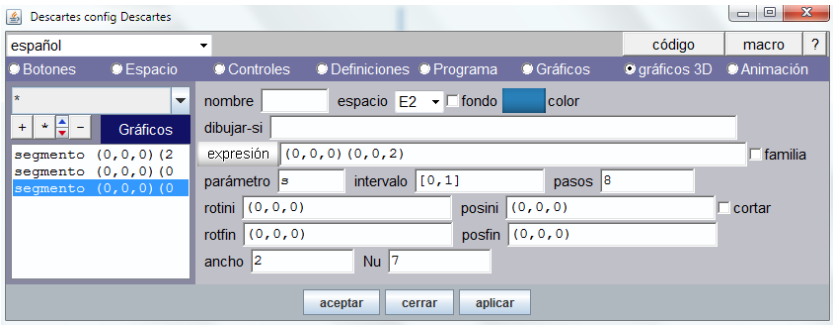

Figura 8.25: Configuración del ejemplo del gráfico tridimensional *Segmento*.

8.2.2. Gráfico *Punto*

Es un gráfico que se inserta sólo en un espacio tridimensional. Es igual al gráfico punto en dos dimensiones, con la salvedad de que el campo *expresión* de éste conlleva tres coordenadas (la primera para el eje x, la segunda para el eje y y la tercera para el eje z).

En la Figura 8.26 se muestra un ejemplo de este tipo de gráfico. En la Figura 8.27 se muestra la configuración necesaria para lograr este ejemplo. Los colores usados para los segmentos son c0392b para el rojo, 27ae60 para el verde y 2980b9 para el azul. Consulte la herramienta de control de colores para mayor información.

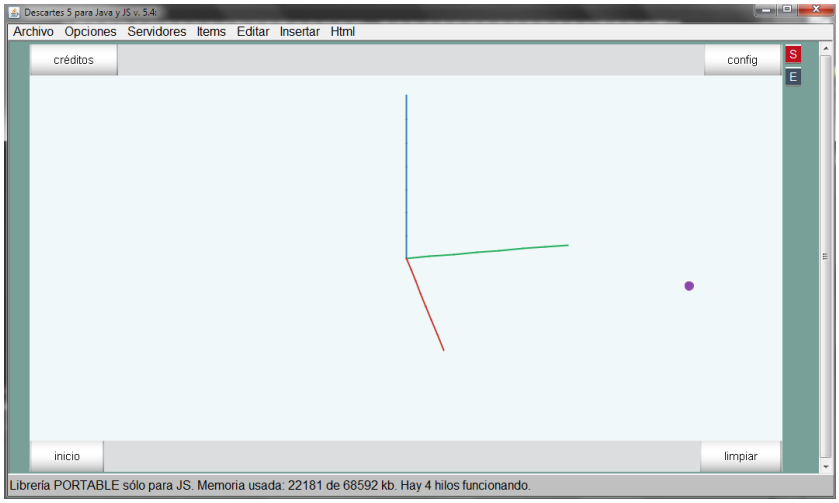

Figura 8.26: Ejemplo del gráfico tridimensional *Punto*.

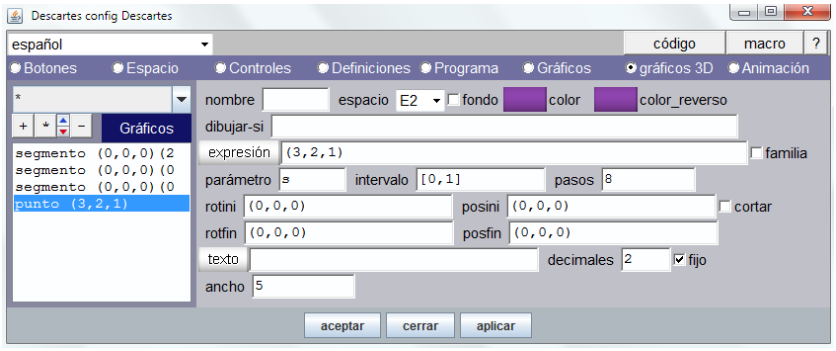

Figura 8.27: Configuración del ejemplo del gráfico tridimensional *Punto*.

8.2.3. Gráfico *Polígono*

Es parecido al gráfico polígono en dos dimensiones, con la salvedad de que las coordenadas de los vértices en el campo *expresión* de éste conllevan tres entradas cada una (una entrada por cada eje en tres dimensiones).

En la Figura 8.28 se muestra un ejemplo de este tipo de gráfico. En la Figura 8.29 se muestra la configuración necesaria para lograr este ejemplo. Note que en este ejemplo, los vértices del polígono pueden no necesariamente pertenecer a un mismo plano.

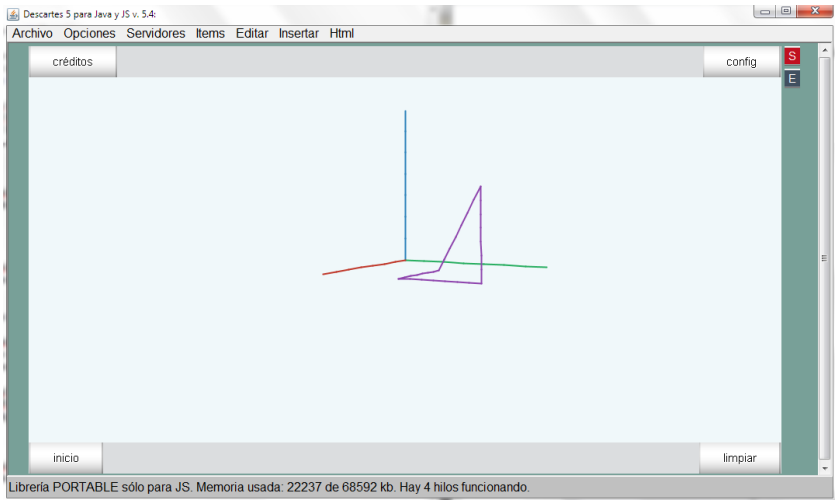

Figura 8.28: Ejemplo del gráfico tridimensional *Poligono*.

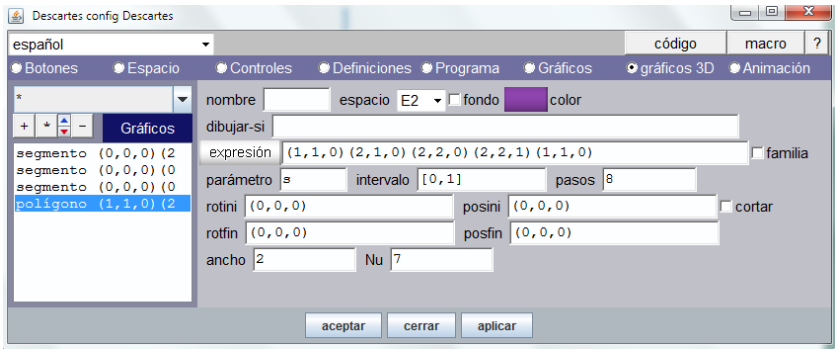

Figura 8.29: Configuración del ejemplo del gráfico tridimensional *Poligono*.

8.2.4. Gráfico *Curva*

Es parecido al gráfico curva en dos dimensiones, con la salvedad de que el campo *expresión* de éste lleva una expresión para x, una para y y una para z cada una separada de la otra por un espacio. Las expresiones dependen de un parámetro u que adopta valores continuos entre 0 y 1.

En la Figura 8.30 se muestra un ejemplo de el gráfico tridimensional curva. En la Figura 8.31 se muestra la configuración para lograr dicho ejemplo.

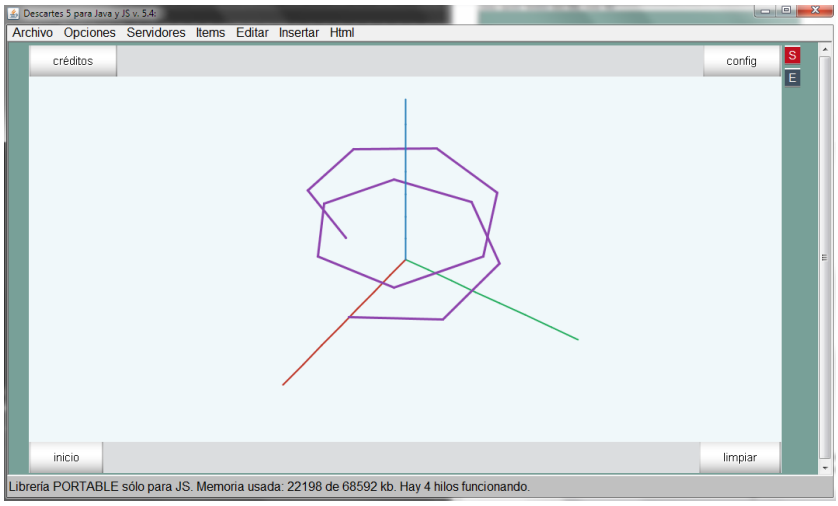

Figura 8.30: Ejemplo del gráfico tridimensional *Curva*.

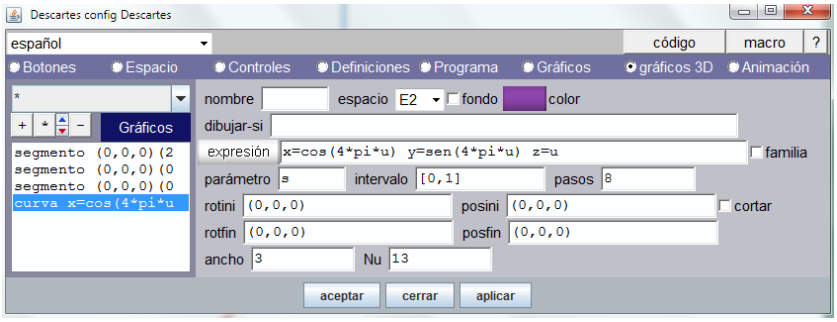

Figura 8.31: Configuración del ejemplo del gráfico tridimensional *Curva*.

Incluye un campo *Nu* en el que se introduce el número de partes en que se subdivide el intervalo entre 0 y 1 que define los valores que adopta el parámetro *u*. Mientras más grande sea este valor, más fina será la curva, pero hacer éste valor demasiado grande puede resultar en ralentizar el interactivo.

8.2.5. Gráfico *Triángulo*

Es básicamente un polígono de tres lados solamente. Puede referirse al gráfico polígono tridimensional para mayor información.

En la Figura 8.32 se muestra un ejemplo de este gráfico. En la Figura 8.33 se muestra la configuración para conseguir este ejemplo.

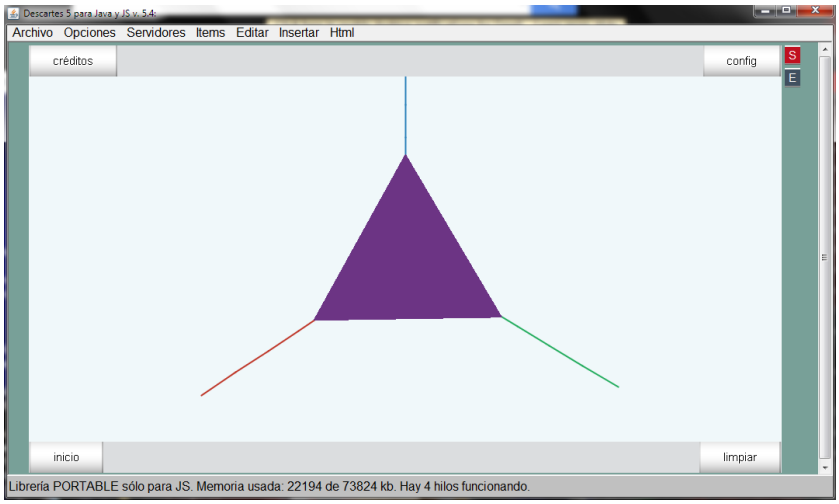

Figura 8.32: Ejemplo del gráfico *Triángulo*.

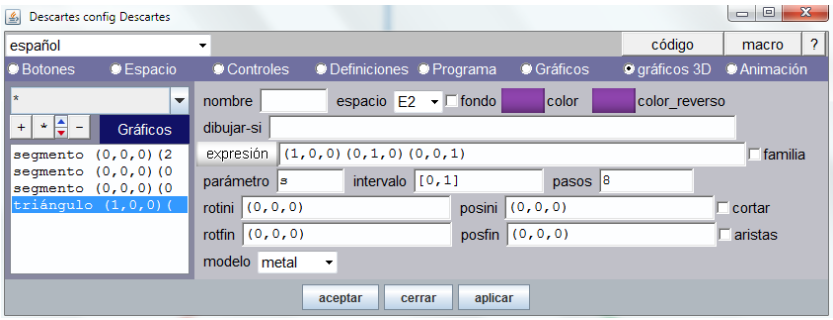

Figura 8.33: Configuración del ejemplo del gráfico *Triángulo*.

8.2.6. Gráfico *Cara*

Permite definir en su campo *expresión* los vértices de una cara en dos dimensiones (cada vértice se asocia a un par ordenado, no a tres coordenadas). Posteriormente es posible definir la inclinación tridimensional de la cara creada usando los campos *rotini*, *posini*, *rotfin* y *posfin*. Para mayor información sobre estos campos consulte el apartado sobre los controles comunes a los gráficos 3D.

En la Figura 8.34 se muestra un ejemplo de este gráfico. En la Figura 8.35 se muestra la configuración que ha de implementarse para reproducir este ejemplo. Note que la cara en este ejemplo está originalmente definida en el plano XY ya que sólo se usan pares ordenados y no triadas. No obstante, echando mano de una rotación en el parámetro *rotini*, esta cara puede mostrarse con una inclinación.

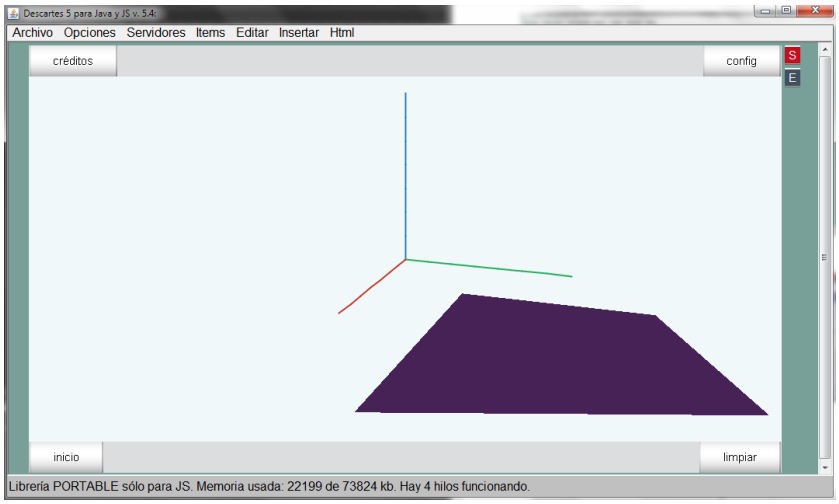

Figura 8.34: Ejemplo del gráfico *Cara*.

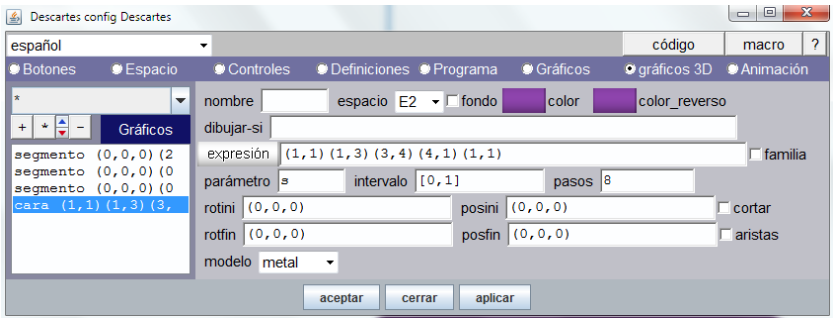

Figura 8.35: Configuración del ejemplo del gráfico *Cara*.

8.2.7. Gráfico *Polireg*

Es un gráfico que permite definir un polígono regular usando un parámetro *Nu* que determina el número de caras del polígono. Permite rotaciones y traslaciones iniciales y finales como se describe en el apartado sobre controles comunes a los gráficos 3D.

En la Figura 8.36 se muestra un ejemplo de este gráfico. En la Figura 8.37 se muestra la configuración para lograr este ejemplo. Note que en este ejemplo el número de lados del polígono regular está determinado por *Nu*. De forma natural, el polígono yacería sobre el plano XY, pero gracias a los cambios en los parámetros *rotini* y *posini*, el polígono se encuentra rotado y parece que se recarga en el eje Z.

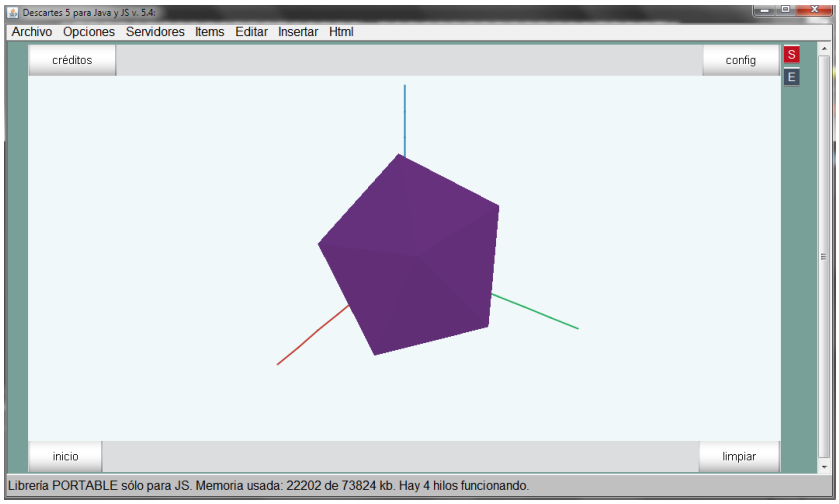

Figura 8.36: Ejemplo del gráfico *Polireg*.

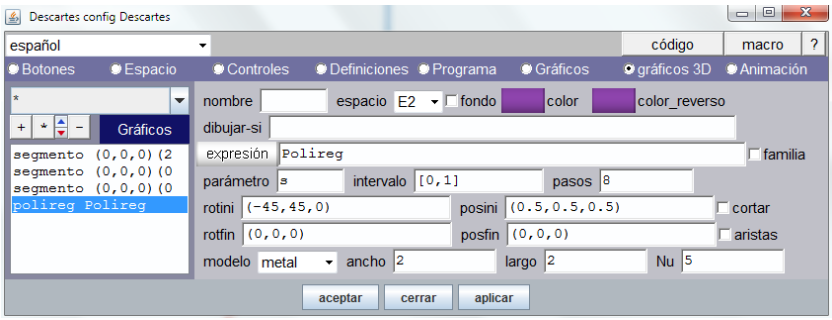

Figura 8.37: Configuración del ejemplo del gráfico *Polireg*.

8.2.8. Gráfico *Superficie*

Es un gráfico que permite definir una superficie a partir de dos parámetros que adoptan valores continuos en el intervalo entre 0 y 1. Los parámetros son u y v. El número de particiones de estos parámetros en su intervalo están dados por los campos Nu y Nv, respectivamente.

Es necesario llenar el campo *expresión* del gráfico con una expresión para x (en función de u y v), una para y y una para z separadas por espacios.

En la Figura 8.38 se muestra un ejemplo de este gráfico. En la Figura 8.39 se muestra la configuración para lograr este ejemplo.

Las asignaciones dadas en este ejemplo en el parámetro *expresión* son $x = 3cos(2\pi u)sin(\frac{pi}{2}v)$, $y = 3sin(2\pi u)sen(\frac{pi}{2}v)$, y $z = cos(3\sqrt{x^2+y^2})$. Note que las coordenadas en este caso son cilíndricas y con eso se logra que el borde de la superficie sea circular. La raíz en la expresión para z involucra la distancia del eje Z, y es el argumento de un coseno, lo que da la sensación de onda en e gráfico.

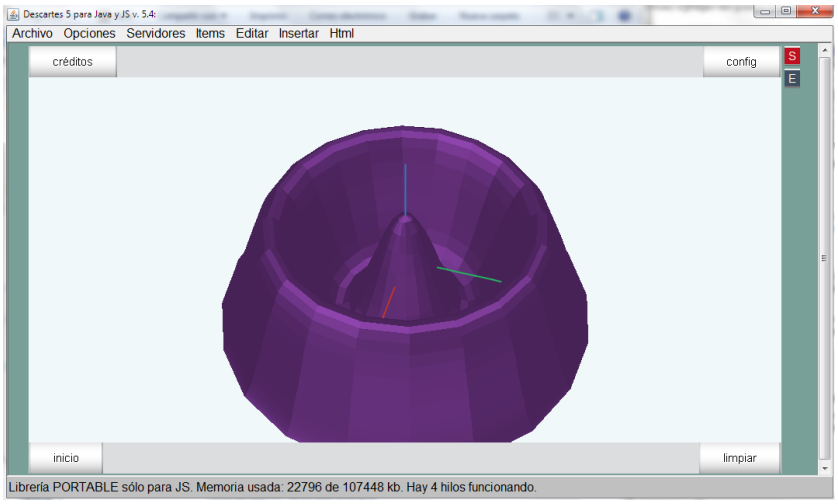

Figura 8.38: Ejemplo del gráfico *Superficie*.

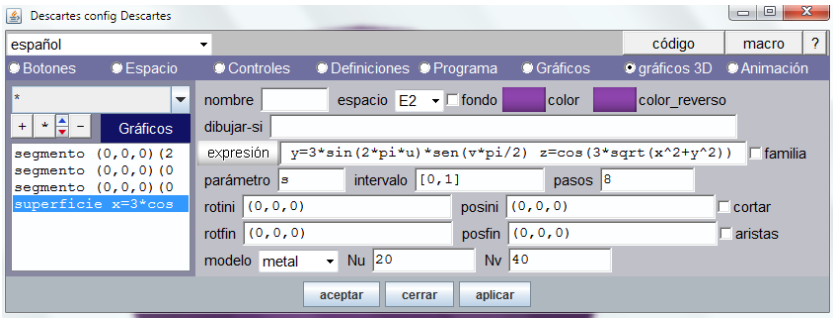

Figura 8.39: Configuración del ejemplo del gráfico *Superficie*.

8.2.9. Gráfico *Texto*

Es igual al gráfico *texto* para espacios en dos dimensiones. No obstante, es posible definirle una rotación y posición finales como se describe en el apartado sobre controles comunes a los gráficos 3D.

8.2.10. Gráfico *Cubo*

Es un gráfico al que se le define un campo *ancho* que corresponde al lado del cubo. Se puede reorientar y trasladar usando los campos *rotini*, *posini*, *rotfin* y *posfin*descritos en el apartado sobre controles comunes a los gráficos 3D.

En la Figura 8.40 se muestra un ejemplo de este gráfico. En la Figura 8.41 se muestra la configuración para lograr este ejemplo. Note que en este ejemplo el cubo aparece rotado debido a la expresión introducida en el parámetro *rotini*.

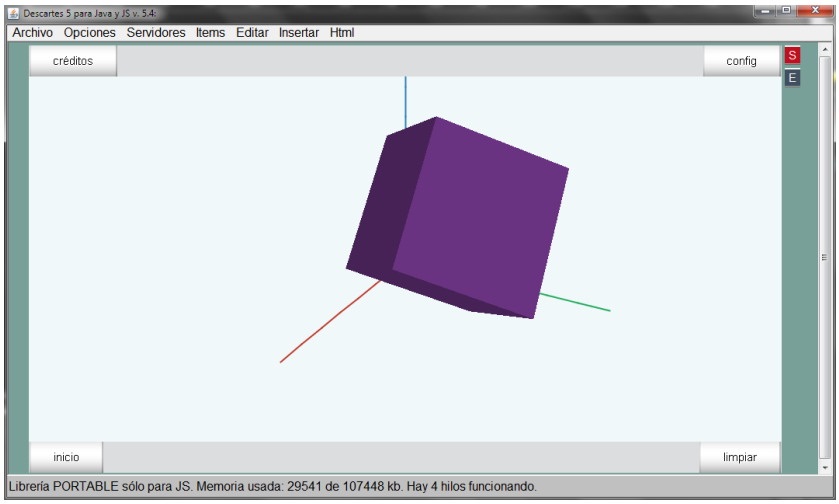

Figura 8.40: Ejemplo del gráfico *Cubo*.

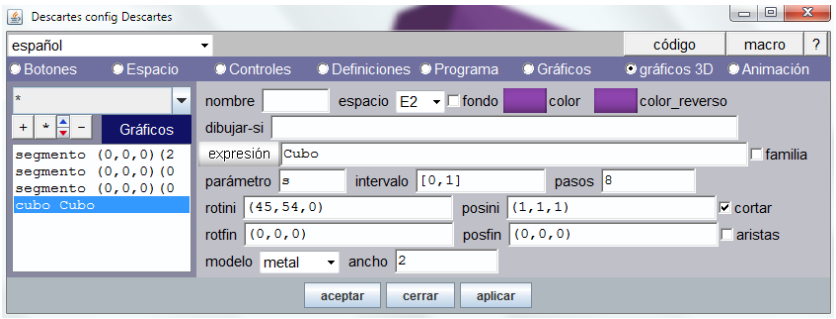

Figura 8.41: Configuración del ejemplo del gráfico *Cubo*.

8.2.11. Gráfico *Paralelepípedo*

Es un gráfico parecido al cubo, salvo que a éste se le define un ancho, largo y un alto.

En la Figura 8.42 se muestra un ejemplo de este gráfico. En la Figura 8.43 se muestra la configuración para lograr este ejemplo. Note que en este ejemplo el cubo aparece rotado debido a la expresión introducida en el parámetro *rotini*.

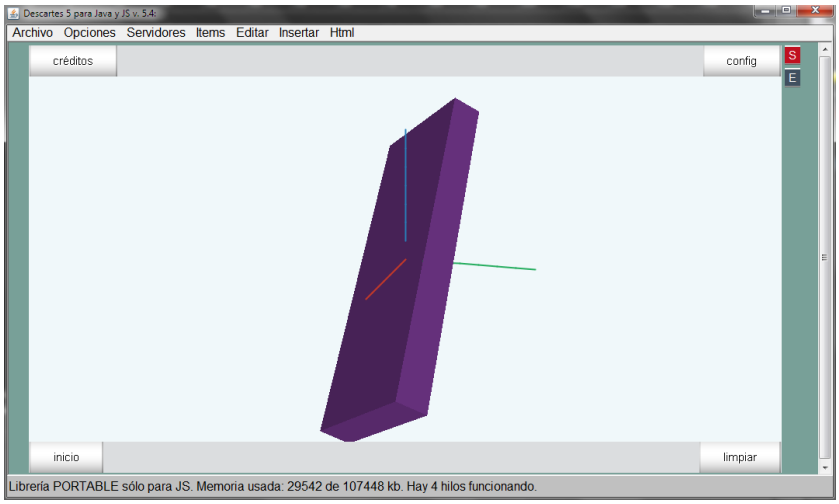

Figura 8.42: Ejemplo del gráfico *Paralelepípedo*.

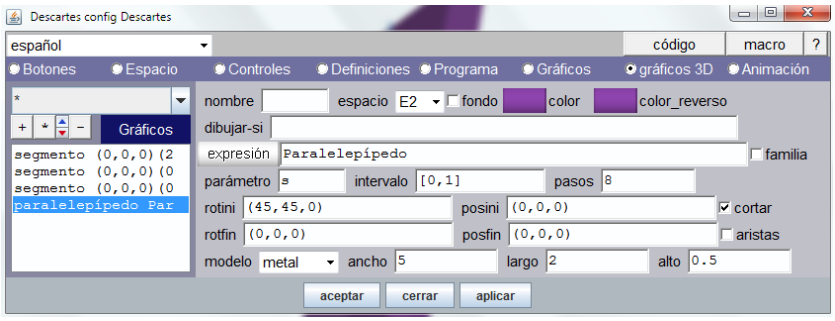

Figura 8.43: Configuración del ejemplo del gráfico *Paralelepípedo*.

8.2.12. Gráficos *Tetraedro*, *Octaedro*, *Dodecaedro* e *Icosaedro*

Son gráficos prediseñados a los que se les asigna un cierto ancho que determina sus tamaños. Se pueden reorientar y trasladar usando los campos *rotini*, *posini*, *rotfin* y *posfin*descritos en el apartado sobre controles comunes a los gráficos 3D.

En la Figura 8.44 se muestra un ejemplo de uno de estos gráficos: el icosaedro. En la Figura 8.45 se muestra la configuración para lograr este ejemplo. Note que en este ejemplo el icosaedro no aparece centrado en el origen debido a que la expresión introducida en el parámetro *rotini* desplaza el centro del icosaedro a esa posición.

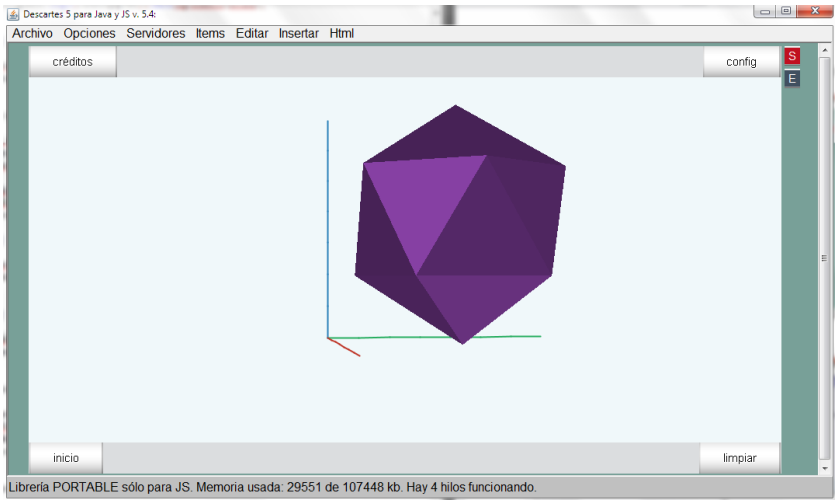

Figura 8.44: Ejemplo del gráfico *Icosaedro*.

Figura 8.45: Configuración del ejemplo del gráfico *Icosaedro*.

8.2.13. Gráfico *Esfera*

Es un gráfico al que se le asigna un valor para un parámetro *ancho* que define su diámetro, así como valores para su campo *Nu* (que define el número de paralelos para la división de la esfera) y su campo *Nv* (que define el número de meridianos para la división de la esfera). Mientras mayores sean *Nu* y *Nv*, el gráfico se verá más parecido a una esfera, pero exagerar el valor de estos parámetros puede resultar en un interactivo de lenta actualización.

En la Figura 8.46 se muestra un ejemplo de este gráfico. En la Figura 8.47 se muestra la configuración para lograr este ejemplo.

Note que en este ejemplo los paralelos y meridianos de la esfera se marcan de un tono que resalta. Ello es porque el checkbox *aristas* se encuentra marcado. Observe que hay muchos paralelos y pocos meridianos. Ello corresponde a que el parámetro *Nu* vale 15 y el *Nv* vale 7. También vea que los paralelos de la esfera no son horizontales respecto al plano XY (aquél marcado por el eje rojo y el verde) del espacio. Ello responde a que hay una expresión, usando ángulos de Euler, introducida en el parámetro *rotini* que se encarga de rotar la esfera.

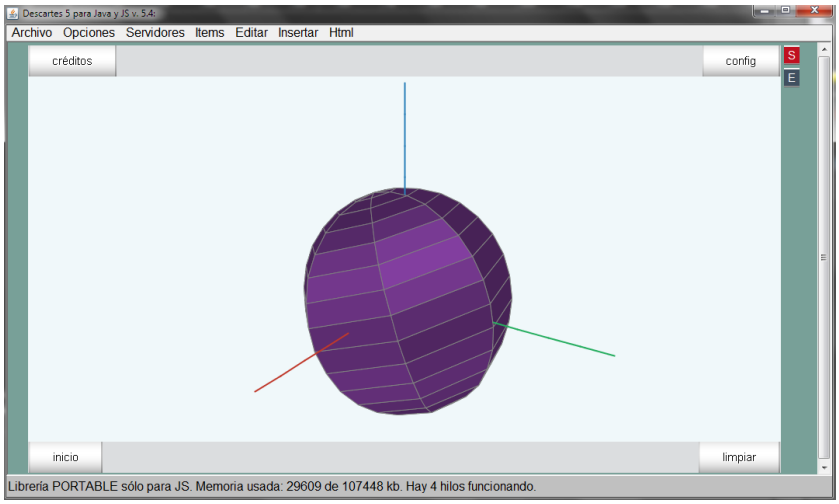

Figura 8.46: Ejemplo del gráfico *Esfera*.

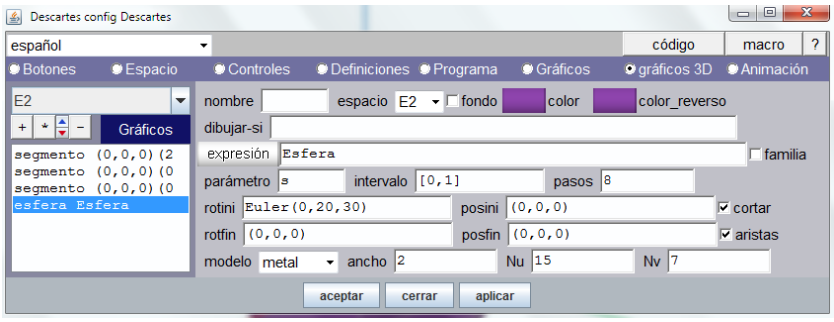

Figura 8.47: Configuración del ejemplo del gráfico *Esfera*.

8.2.14. Gráfico *Elipsoide*

Es un gráfico al que se le asigna un valor para los campos *ancho*, *largo* y *alto*, que corresponden a las longitudes de los ejes en x, y y z del elipsoide, respectivamente. También involucra un parámetro Nu y uno Nv con las que se determina el número de secciones en paralelos y en meridianos, respectivamente.

En la Figura 8.48 se muestra un ejemplo de este tipo de figura. En la Figura 8.49 se muestra la configuración necesaria para lograr este ejemplo.

Note que el elipsoide tiene cierta transparencia para poder ver los segmentos que representan los ejes detrás de él. Las aristas se encuentran marcadas para que se vea el efecto de los parámetros Nu y Nv.

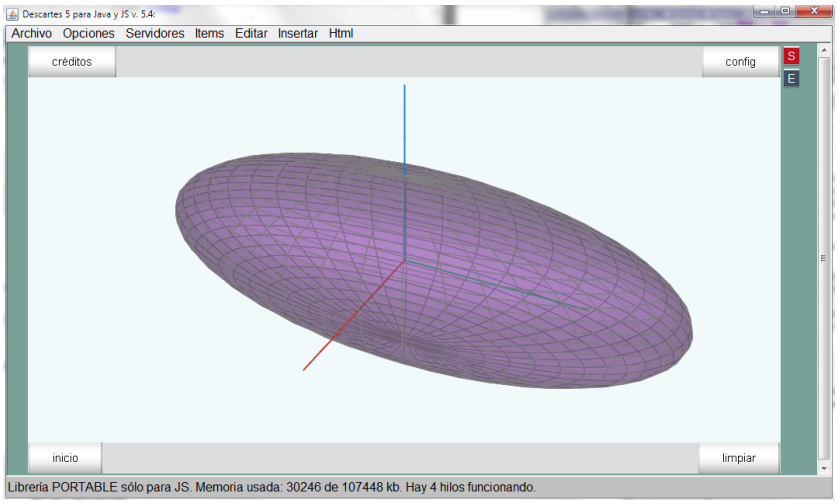

Figura 8.48: Ejemplo del gráfico *Elipsoide*.

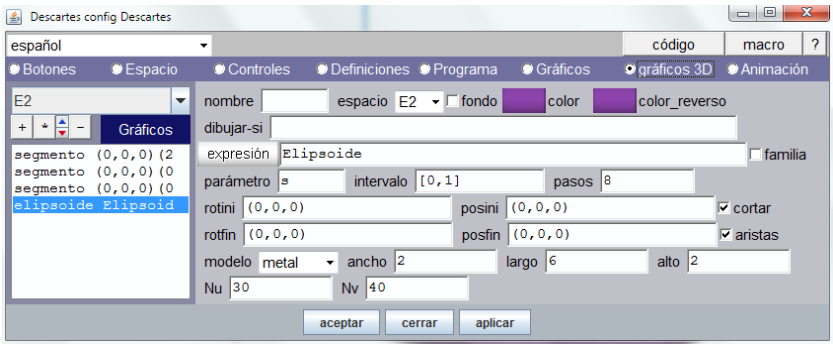

Figura 8.49: Configuración del ejemplo del gráfico *Elipsoide*.

8.2.15. Gráfico *Cono*

Es un gráfico de forma cónica cuya base puede ser una elipse. A este gráfico se le define un parámetro *ancho*, que corresponde al eje en x de la elipse que es la base; uno *largo*, que define el eje en y de la elipse de base; y uno *alto* que define la altura en el eje z del cono. También cuenta con un parámetro *Nu* que determina el número de vértices que definen la curva que forma la elipse de la base. Este número es, así pues, algo así como los meridianos que definen el gráfico (semejantes a los meridianos de una esfera). Cuenta también con un parámetro *Nv* que, aunque no brinda un cambio en la resolución del gráfico como en otras figuras tridimensionales, define el número de paralelos (también semejantes a los de una esfera) del gráfico.

En la Figura 8.50 se muestra un ejemplo de este tipo de gráfico. En la Figura 8.51 se muestra la configuración necesaria para lograr este ejemplo.

Note que las aristas se encuentran marcadas para que se vea el efecto de los parámetros *Nu* y *Nv*. Observe que el número de paralelos del gráfico (curvas paralelas al plano XY) es 10, como lo define el parámetro *Nv*. También note que el gráfico se encuentra rotado 180°respecto al eje X debido a la expresión en el parámetro *rotini*. Es decir, de forma natural, el vértice del cono aparece hacia abajo, pero aquí se rotó para que la base aparezca abajo. Asimismo, el centro del gráfico se encuentra desplazado del origen debido a la expresión en el parámetro *posini*.

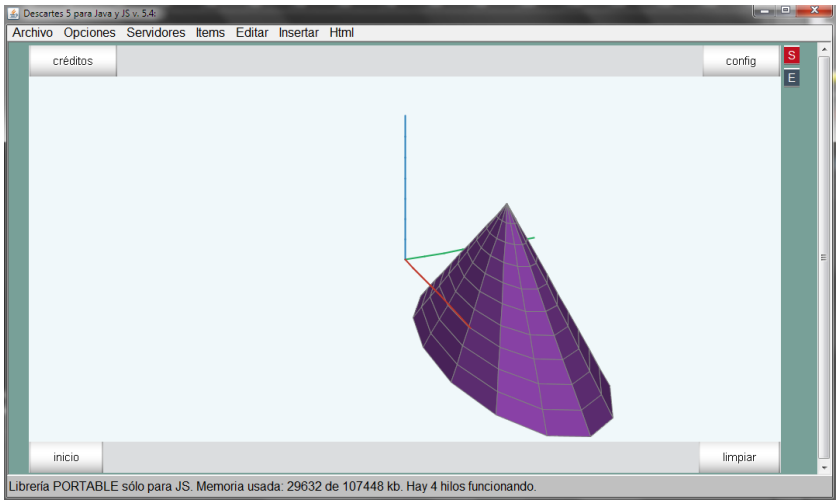

Figura 8.50: Ejemplo del gráfico *Cono*.

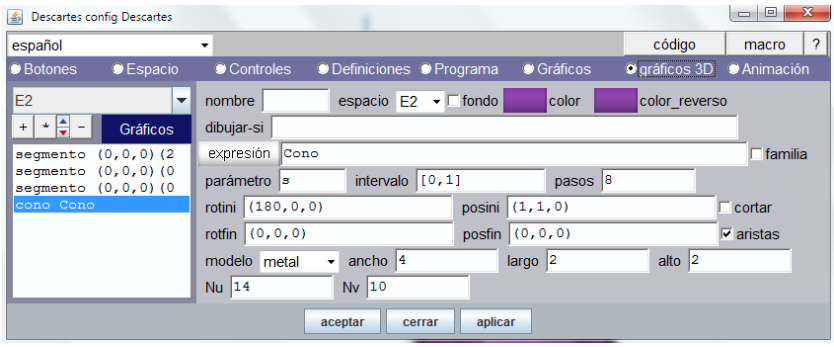

Figura 8.51: Configuración del ejemplo del gráfico *Cono*.

8.2.16. Gráfico *Cilindro*

Es un gráfico de forma cilíndrica cuya base puede ser una elipse. Cuenta con un parámetro *ancho*, que corresponde al eje en x de la elipse que es la base; uno *largo*, que define el eje en y de la elipse de base; y uno *alto* que define la altura del cilindro en el eje z. También cuenta con un parámetro *Nu* que determina el número de vértices que definen la curva que forma la elipse de la base. Este número es, así pues, algo así como los meridianos que definen el gráfico (semejantes a los de una esfera). Cuenta también con un parámetro *Nv* que, aunque no brinda un cambio en la resolución del gráfico como en otras figuras tridimensionales, define el número de paralelos (también semejantes a los de una esfera) del gráfico.

En la Figura 8.52 se muestra un ejemplo de este tipo de gráfico. En la Figura 8.53 se muestra la configuración necesaria para lograr este ejemplo.

Note que las aristas se encuentran marcadas para que se vea el efecto de los parámetros *Nu* y *Nv*. También note que el gráfico se encuentra rotado 45°respecto al eje X y 45°respecto al Y (observe la expresión en su parámetro *rotini*), por lo que no aparece vertical o parado.

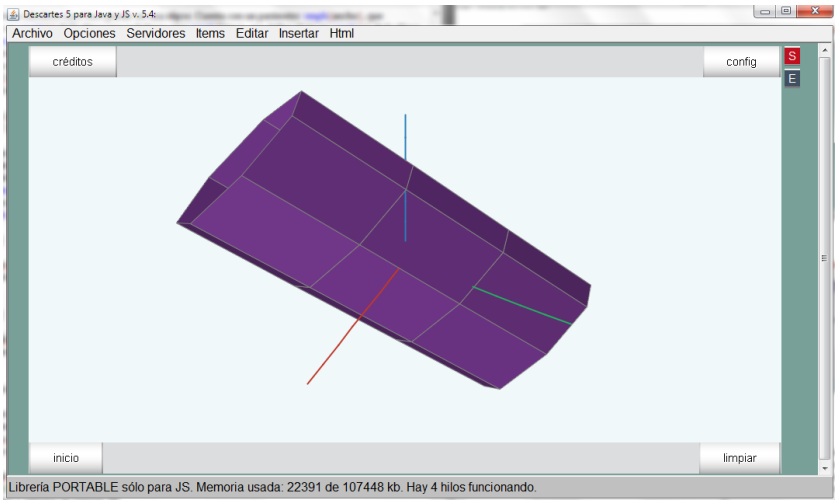

Figura 8.52: Ejemplo del gráfico *Cilindro*.

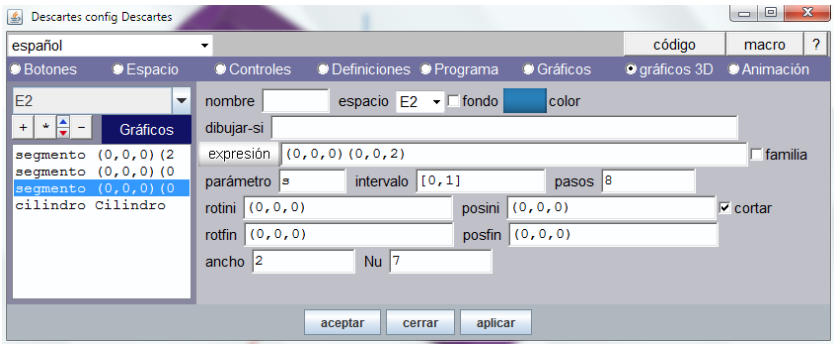

Figura 8.53: Configuración del ejemplo del gráfico *Cilindro*.

8.2.17. Ejercicio general de gráficos 3D

Realicemos un ejercicio general sobre gráficos 3D. Elegimos uno representativo, que es el elipsoide, dado que su funcionamiento permite entender el de otros gráficos 3D. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Gráficos3D. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Graficos3D/Graficos3D_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Este ejercicio sirve como una muestra de la funcionalidad general de los gráficos tridimensionales. En particular, el uso de las rotaciones y posiciones iniciales y finales permite reducir la cantidad de cálculos para lograr un gráfico con una determinada disposición. Por ejemplo, sería posible incluir un polígono inclinado y desplazado dándolo sólo sus vértices. No obstante, esto involucraría muchos cálculos que se pueden ahorrar trazando el polígono en el origen y luego sólo indicando sendas rotaciones y traslaciones. Es muy importante tomar en cuenta el orden en que se hacen las rotaciones y traslaciones para lograr una posición e inclinación particular de un gráfico.

Es importante enfatizar el uso de segmentos para no perder de vista en qué dirección apuntan los ejes del espacio tridimensional. Estos segmentos suelen sólo ser útiles al programador mientras coloca sus gráficos, por lo que pueden finalmente ser ocultados.

Se sugiere que el usuario experimente un poco más agregando algunas otras figuras y pruebe el funcionamiento del checkbox *cortar* cuando éstas se intersecan. Igualmente, conviene que vea los cambios que el menú *modelo* genera en los gráficos.

Para este ejercicio se cambiaron los colores de los ejes. Puede consultar el apartado sobre la herramienta de control de colores para más información.

8.3. Controles comunes a los gráficos

- **menú de espacios**: es un menú que muestra los espacios existentes en el interactivo. A diferencia del menú de espacios que se encuentra sobre el panel a la izquierda, este menú selecciona adónde mover el gráfico seleccionado, en vez de filtrar los gráficos de un espacio como hace el otro menú. Se recomienda tener en mente esta diferencia pues se pueden hacer errores si se usa un menú en lugar del otro.
- **fondo**: es un checkbox que si está seleccionado hace que el gráfico en cuestión se dibuje una sola vez al cargar el interactivo. Si se encuentra marcado, los cambios subsecuentes en variables que controlen a dicho gráfico (su color, forma, etc.) no se reflejarán en el mismo. Es decir, el gráfico se encuentra como un fondo estático (no interactivo) de la escena. Este checkbox suele marcarse cuando el gráfico en cuestión se planea que no cambie. Esto permite que el interactivo cargue más rápidamente y no se perderá tiempo refrescando el gráfico constantemente.
- **dibujar-si**: es un campo de texto en el cual se introduce alguna condición booleana que determina si el gráfico ha de pintarse o no. Una condición booleana puede ser verdadera (a la que se le otorga un valor de 1) o falsa (a la que se le otorga un valor de 0). Por ejemplo, si se tuviera una variable a con valor de 2 y en el campo de texto se introduce la condición $a == 2$, se compara el valor de a con 2 y se ve que no son iguales, por lo que la condición vale 0 y el gráfico no se pintará. Si a valiera 2, entonces sí se pintaría. Más adelante se verán más ejemplos de condiciones booleanas. Es importante tener en cuenta que el comportamiento de este campo de texto no sólo aplica a los gráficos, sino que también puede ser usado en controles, como se verá después.
- **coordenadas absolutas**: es un checkbox que determina si el gráfico se pinta en coordenadas absolutas o en relativas. En coordenadas absolutas, el origen está en la esquina superior izquierda del espacio, las unidades se miden en pixeles y los valores de las ordenadas crecen hacia abajo. En coordenadas relativas, el origen es el origen del plano cartesiano del espacio, las unidades son las determinadas por la escala del espacio, y el valor de las ordenadas crece hacia arriba.
- **rastro**: es un checkbox acompañado de un control de color. Cuando el gráfico en cuestión se modifica dinámicamente (por ejemplo, cambia su forma o posición), éste simplemente se borra y se traza en su nueva forma o posición. Pero si este checkbox está marcado, dejará rastros de sus posiciones y formas anteriores conforme cambia. El control de color permite elegir qué color tiene el rastro dejado por el gráfico.
- **familia**: es un checkbox que determina si el gráfico ha de pintarse de forma única, o si ha de repetirse su trazo varias veces. El número de veces y las posiciones de trazo de una familia están determinadas por los controles *parámetro*, *intervalo* y *pasos* descritos a continuación.
- **parámetro**: es un campo de texto en el que se introduce una variable o letra cuyo valor varía de acuerdo al número de pasos que caben en un intervalo dado. El número de pasos e intervalo se describen a continuación.
- **intervalo**: es un campo de texto en el que se introduce un intervalo entre corchetes cuadrados. El intervalo consiste en dos números separados por una coma. El primer número es el primer valor que tendrá la variable dada en *parámetro*, mientras que el segundo número será el último valor que tiene. Este intervalo se divide en $n - 1$ partes iguales (donde n es el número de pasos elegido) de tal forma que adopta valores equidistantes uno del siguiente dentro del intervalo. El número de pasos usado es el que se detalla a continuación.

- **pasos**: es un campo de texto donde se indica el número de pasos que da la variable usada en *parámetro* de tal forma que, empezando en el valor izquierdo del intervalo, salta de forma equidistante ése número de pasos y termina en el valor derecho del intervalo.
- **ancho**: es un campo de texto en el que se indica el número de pixeles de grosor del gráfico en cuestión.
- **info**: es un campo de texto donde suele introducirse una descripción del gráfico. Es una buena práctica introducir información relevante en este control para fácilmente ubicar los gráficos según su función, sobre todo cuando hay un gran número de gráficos. Este control no tiene efecto en el interactivo, y sólo sirve para identificar más fácilmente un gráfico dado.

8.4. Controles comunes a los gráficos 3D

A continuación se presenta una descripción de los controles comunes a estos gráficos. No se presenta una descripción de todos; sólo de aquellos particulares a los gráficos 3D y que no están incluidos en los controles comunes a los gráficos de dos dimensiones.

- **color_reverso**: es un botón que lanza la herramienta de control de colores. Para ciertos gráficos tridimensionales, además del color común que se define también para los gráficos bidimensionales, hay un color reverso que es el de la cara posterior del gráfico. Es éste el que se define mediante el control en cuestión. Para más información sobre la funcionalidad de la herramienta que se lanza con este botón, consulte el apartado sobre la herramienta de control de colores.
- **rotini**: es un campo de texto en el que se introduce la rotación inicial, en forma de un vector de tres entradas. Una vez que se define un gráfico mediante el campo *expresión* en el selector *gráficos 3D*, éste se puede rotar de forma inicial con el campo en cuestión. La primera entrada del vector corresponde a una rotación del gráfico alrededor del eje x. La segunda entrada corresponde a una rotación en grados respecto al eje y y la tercera respecto al eje z.
  En ocasiones puede usarse una rotación con ángulos de Euler, en la la primera entrada corresponde a una rotación en grados respecto al eje z, con lo que el eje x ya no es el mismo. La segunda entrada corresponde a una rotación en grados respecto al nuevo eje x, con lo que el eje z habrá cambiado. Y la tercera entrada corresponde a una rotación en grados respecto al nuevo eje z. Para usar este tipo de ángulos es necesario incluir la palabra *Euler* inmediatamente antes del vector.
- **posini**: es un campo de texto en el que se introduce un vector de tres entradas que corresponde a la traslación inicial del gráfico. La primera entrada corresponde a la traslación en el eje x, la segunda a aquella en el eje y y la tercera a aquella en el eje z. Esta traslación se aplica después de haber hecho la rotación inicial.
- **rotfin**: es un campo de texto que funciona igual que el *rotini*, pero que se aplica después de haber hecho la traslación inicial definida por el campo *posini*.
- **posfin**: es un campo de texto que funciona igual que el *posini*, pero que se aplica después de haber hecho la rotación final definida por el campo *rotfin*.
- **cortar**: es un checkbox que cuando está marcado permite que los gráficos que se intersecan en el espacio sean pintados considerando estas intersecciones. De esta forma se hace notoria la intersección entre gráficos, que de lo contrario puede no pintarse correctamente. Dado que esta opción puede ralentizar al interactivo, sólo debe activarse cuando no hay muchos gráficos y no hay intersecciones entre ellos.
- **aristas**: es un checkbox que cuando está marcado hace que se pinten las aristas del gráfico en cuestión (que pueden ser caras, polígonos y superficies). El color con que se pintan es gris.
- **modelo**: es un menú con cuatro opciones:

color: se usa para pintar el gráfico con un color fijo.

luz: se usa para dar sensación de iluminación. De tal forma que el gráfico puede ser más o menos brillantes en algunas partes dependiendo de su orientación.

metal: se usa para dar sensación de iluminación igual que la opción *luz*, pero aumenta el contraste de los brillos, con lo que la superficie aparenta ser metálica.

alambre: se usa para sólo trazar las orillas del gráfico con el color seleccionado.

- **Nu**: es un campo de texto en donde se introduce un número entero que determina, para los gráficos que dependen de un parámetro u, en cuántas partes se dividirá el intervalo [0,1] que da sus valores a u.
- **Nv**: es un campo de texto que funciona igual que el *Nu*. Aplica a los gráficos que requieren dos parámetros para ser trazados, como el gráfico *superficie*.
- **ancho**: es un campo de texto que aplica a algunos gráficos como emphpolireg que indica el ancho del mismo.
- **largo**: es un campo de texto que aplica a algunos gráficos como emphpolireg que indica el largo del mismo.

Capítulo 9

El selector *Controles*

Ya que hemos analizado el selector *Gráficos*, nos conviene regresar al de controles. En algunas ejercicios ya se ha visto un poco sobre los controles numéricos. En este apartado se abordarán todos los controles de forma profunda.

Los controles son objetos con los que el usuario, valga la redundancia, interactúa con el interactivo. Éstos permiten que el usuario haga modificaciones a los valores de las variables, lance animaciones, ejecute funciones, así como y un gran número de otras acciones. Muchas de estas herramientas de interacción son seguramente familiares para el usuario, como lo son los pulsadores, las barras horizontales, los menús y los botones.

Al igual que en el caso de los gráficos ya abordados, hay muchos elementos en común entre los diferentes controles. Se estudiarán sólo los elementos particulares a cada control, y al final de este apartado se estudiarán los elementos en común a todos los controles.

El selector *controles* se escoge marcando las opción *controles* hasta arriba del editor de configuraciones. En la Figura 9.1 se muestra el editor de configuraciones cuando está seleccionado el selector *Controles* justo después de que se añade un control numérico nuevo, que por defecto es un control tipo *pulsador*.

Al igual que con los gráficos, el selector de controles tiene un panel a la izquierda dentro del cual se enlistan los controles existentes en todo el interactivo. Arriba de este panel se encuentran los botones para crear, duplicar, desplazar dentro de la lista y eliminar algún control. Su funcionamiento es igual que en los gráficos, cuya descripción se encuentra en el apartado Gráficos. En la Figura de ejemplo se muestra el selector *Controles* como se muestra por defecto tras añadir un control.

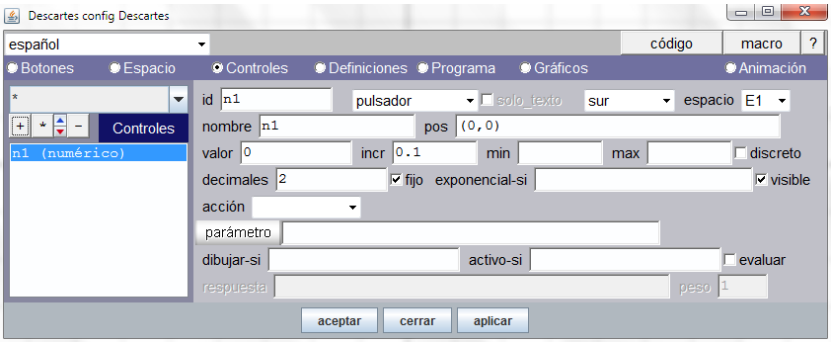

Figura 9.1: Ejemplo del selector *Controles*.

Hay cinco tipos de controles: los *numéricos*, los *gráficos*, el *texto*, *video* y *audio*. Al añadir un nuevo control, aparece una ventana donde se ha de seleccionar en un menú el tipo de control deseado.

9.1. Control numérico tipo *Pulsador*

Este tipo de control numérico es el que se arroja al añadir un control numérico nuevo. Se asocia con una variable con el mismo nombre del identificador del pulsador. El pulsador permite aumentar y disminuir los valores de la variable en cuestión mediante un botón pequeño superior y uno inferior, respectivamente. Todos los elementos del pulsador se pueden consultar dentro de los elementos comunes a los controles.

En la Figura 9.2 se muestra un ejemplo de un control numérico tipo pulsador en una escena. En la Figura 9.3 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que el pulsador en este ejemplo se supone se puede usar para controlar la escala del espacio *E1*, como se indica en el parámetro de sus cálculos. Observe que la ubicación del control en este ejemplo se usaron variables de espacio, que pueden consultarse en el apartado Variables de Espacio

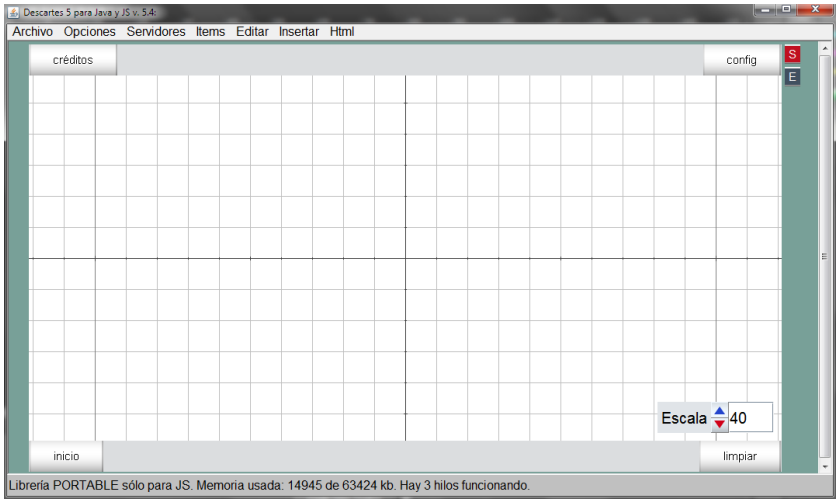

Figura 9.2: Ejemplo del control numérico *Pulsador*.

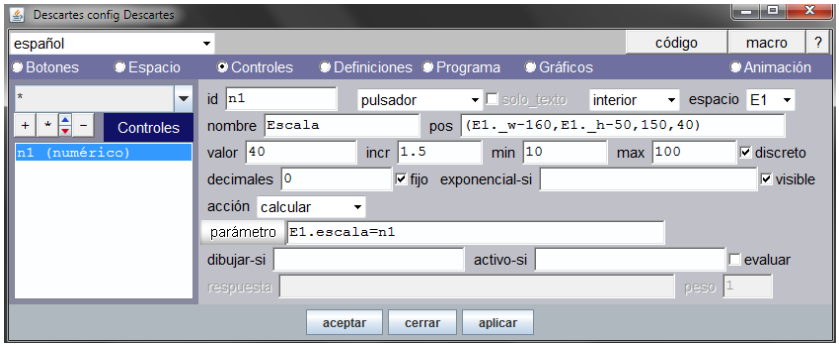

Figura 9.3: Configuración del ejemplo del control numérico *Pulsador*.

Hagamos un breve ejercicio con dos pulsadores que controlan el ancho y alto de un triángulo rectángulo. El interactivo mostrará la longitud de la hipotenusa y el área del triángulo. El interactivo de

este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 01. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Controles_01/Controles_01_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio se pudo observar la importancia del menú *acción* de los controles, así como la forma de introducir el texto en el parámetro del mismo. También se observa la importancia de incluir cálculos iniciales, como los usados en *INICIO* para que el interactivo quede listo con los datos correctos desde el inicio.

9.2. Control numérico tipo *Campo de texto*

Este tipo de control numérico consiste en un campo donde el usuario puede introducir texto o números. Resulta principalmente útil en fines de evaluación. Sus elementos se encuentran descritos en el apartado Elementos comunes a todos los controles.

En la Figura 9.4 se muestra un ejemplo de un control numérico tipo campo de texto en una escena. En la Figura 9.5 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que para este ejemplo el campo de texto se supone se usa para recibir el nombre del usuario. El campo es sólo de texto y aparece vacío de inicio (por eso su valor es un par de comillas sencillas, que es vacío). En la imagen, el usuario ya introdujo su nombre en él. El texto a la izquierda del campo de texto se introdujo mediante la Herramienta de introducción de textos y no tiene que ver directamente con el campo de texto.

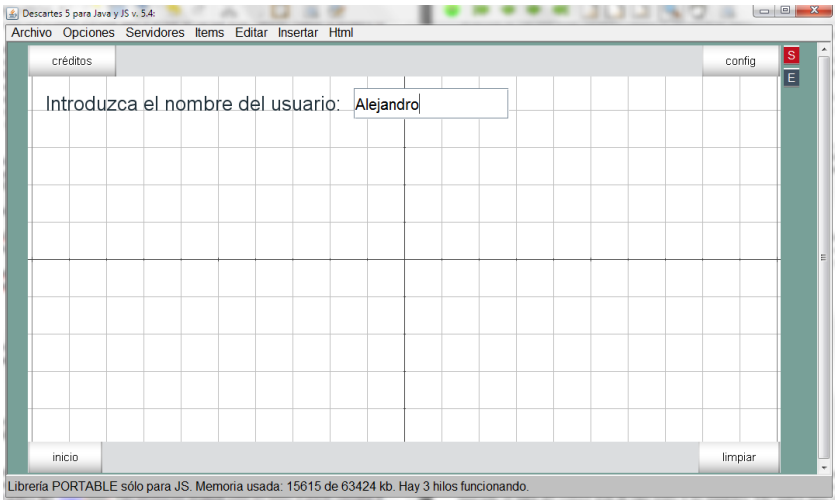

Figura 9.4: Ejemplo del control numérico *Campo de texto*.

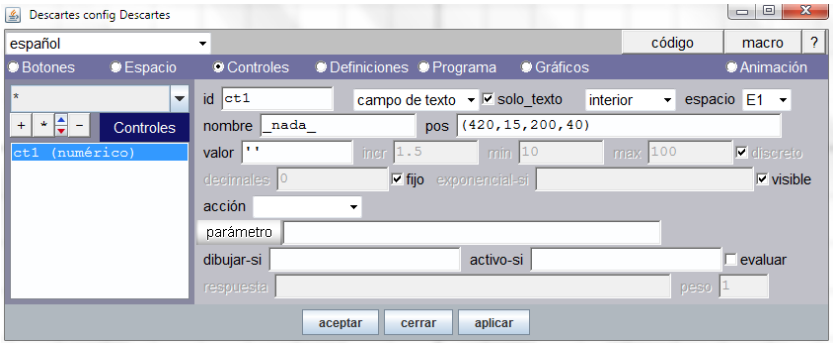

Figura 9.5: Configuración del ejemplo del control numérico *Campo de texto*.

Hagamos un breve ejercicio con un campo de texto donde se muestra cómo se puede manipular lo que el usuario ingresa en él. Se verá la diferencia entre un campo de texto numérico y uno que maneja sólo texto. Aprovecharemos para practicar un poco con el gráfico tipo *texto*. Para información detallada sobre los textos se puede consultar la herramienta de introducción de textos.

El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 02. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Controles_02/Controles_02_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio tuvimos la oportunidad de ver muchas cosas importantes. En particular se vio la diferencia entre variable que llevan texto y las que llevan valores numéricos. El error NaN, además de aparecer cuando se intenta dividir por cero, o cuando se intenta extraer la raíz cuadrada de un número negativo, típicamente también aparece cuando se intenta operar algo que no es un número. Además es importante recordar que porque se le asigne un número a un campo de texto, no necesariamente será considerado como un número, sino que puede tomarse como un caracter o cadena de caracteres.

9.3. Control numérico tipo *Menú*

Este tipo de control numérico es un menú típico. Resulta principalmente útil para permitirle al usuario tener acceso a alguna opción dentro del interactivo. Muchos de los parámetros del menú son comunes a todos los controles y se pueden consultar dentro de los elementos comunes a los controles.

En la Figura 9.6 se muestra un ejemplo de un control numérico tipo menú en una escena. En la Figura 9.7 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que para este ejemplo el menú se supone se usa para recibir una respuesta a una pregunta. El campo tiene 4 opciones, aunque sólo 3 de ellas corresponden a una posible respuesta (las últimas 3). El texto a la izquierda del menú se introdujo usando la Herramienta de introducción de textos y no tiene que ver directamente con el menú.

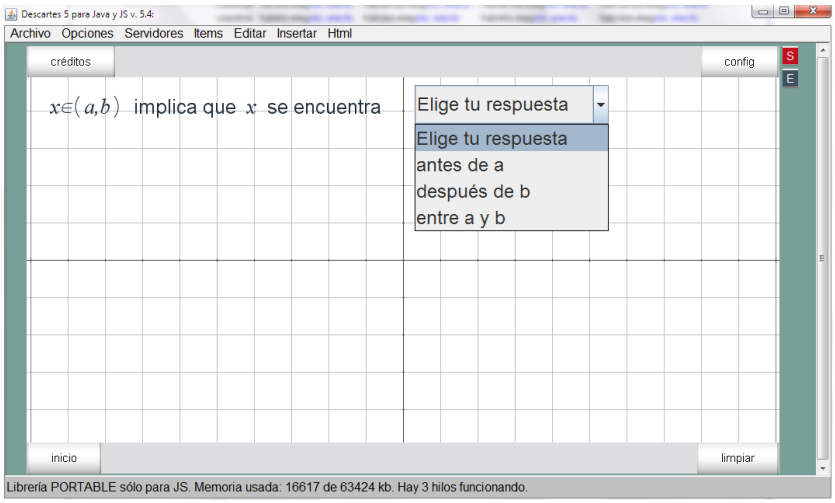

Figura 9.6: Ejemplo del control numérico *Menú*.

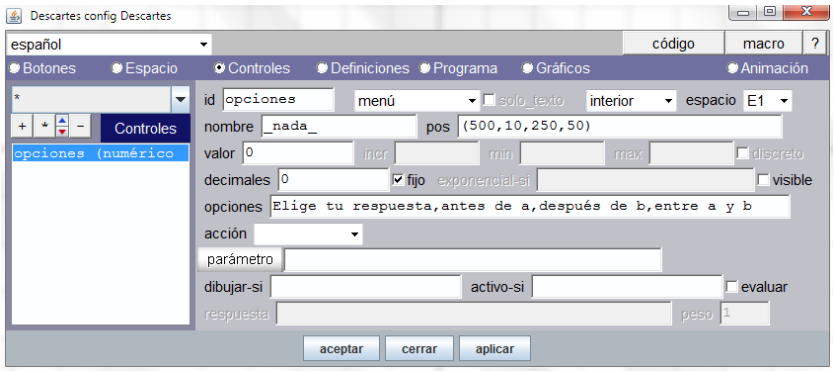

Figura 9.7: Configuración del ejemplo del control numérico *Menú*.

- **valor**: es un campo de texto que indica el valor original que adopta un menú. Para el caso de los menús, que tienen un cierto número de opciones (descritas a continuación), el *valor* debe ser un número entero cero o mayor que indica cuál de las opciones estará seleccionada al iniciar el interactivo. Por ejemplo, un menú con las opciones *sumar*, *restar*, *multiplicar* y *dividir* presentará como opción seleccionada al inicio *multiplicar* si su *valor* esta en 2. Ello se debe a que la primera opción conlleva un valor 0, la segunda un valor 1, la tercera un valor 2 y la cuarta un valor 3.
- **opciones**: es un campo de texto en el que se introducen las opciones del menú. Las opciones se separan mediante comas (,). Si se desea que la primera opción esté vacía, se puede colocar un espacio, luego una coma, y luego el resto de las opciones. Esto se hace cuando se quiere que el menú no tenga una opción particular seleccionada al iniciar el interactivo, sino que aparezca vacío. Las opciones del menú determinan el valor del mismo. La primera opción hace que éste valga 0, la segunda opción hace que valga 1 y así sucesivamente.

Ahora abordaremos un ejercicio que consiste en mostrar distintas figuras y texto dependiendo de la opción del menú seleccionado. Aprovecharemos este ejercicio para practicar, además del menú, algunas otras funcionalidades como las condicionales, algunos gráficos y el uso de espacios para texto. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 03. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/`

Descartes5/desarrollo/doc/Ejercicios/Controles_03/Controles_03_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Este ejercicio es un poco más profesional respecto a los que se habían hecho anteriormente. Por un lado, ya se está tomando en consideración la estética de la escena al centrar el menú y al no mostrar el plano cartesiano que para nuestros propósitos no es necesario. Adicionalmente, el identificador del menú es ahora un nombre escogido por el usuario que permite, valga la redundancia, identificar al control y lo que hace de manera más inmediata. Los gráficos ahora tienen una descripción en su parámetro *info*, lo que permite también identificarlos con más facilidad.

Algo importante que debe notarse es que a veces es conveniente aplicar continuamente los cambios que se hacen. Ello sucede cuando se agrega un espacio nuevo. Si se agrega e inmediatamente se intenta reasignar algún gráfico o control al espacio nuevo, éste aún no aparecerá en la lista de espacios disponibles. Es necesario antes aplicar los cambios para que el espacio nuevo pase a ser parte de la lista.

9.4. Control numérico tipo *Barra*

Este tipo de control numérico consiste en una barra de desplazamiento (o scroll) horizontal que, al moverla, cambia su valor. Puede servir para moverse de un valor a otro alejado de forma rápida, aunque también permite cambios de valores más finos. Todos los parámetros de este control se encuentran entre los elementos comunes a los controles, por lo que no se detalla ninguno en este apartado.

En la Figura 9.8 se muestra un ejemplo de un control numérico tipo barra en una escena. En la Figura 9.9 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que para este ejemplo la barra se supone también se usa para controlar la escala del espacio *E1*, como se ve en los parámetros de su cálculo. Esto no es funcionalidad propia de la barra, pero puede consultarla en el apartado Variables de Espacio.

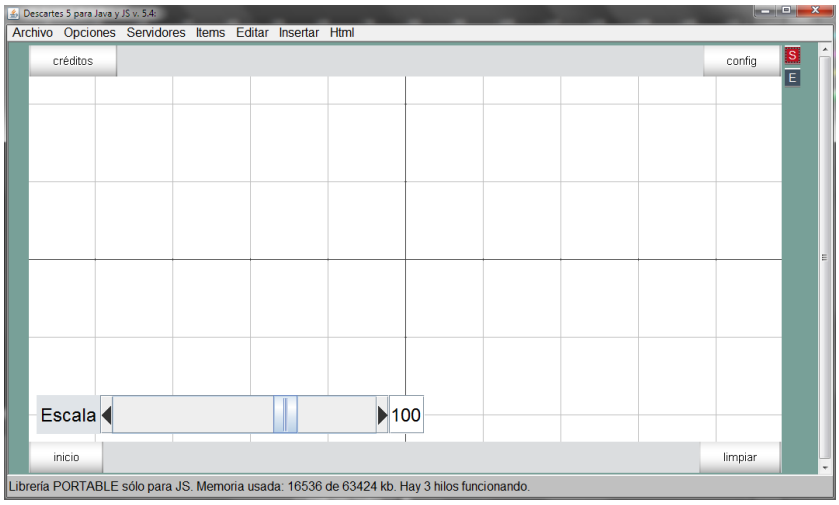

Figura 9.8: Ejemplo del control numérico *Barra*.

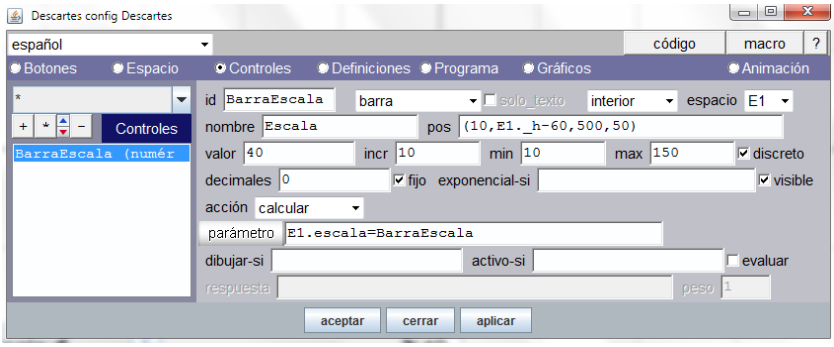

Figura 9.9: Configuración del ejemplo del control numérico *Barra*.

Hagamos un ejercicio que consiste en mostrar un espacio que contiene la gráfica de una ecuación. La idea es que la barra nos sirva como un elemento que controla la escala de este espacio para poder ver particularidades de la gráfica. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 04. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Controles_04/Controles_04_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio pudimos practicar el control de la escala mediante un control numérico. Se abordó también la realización de un cálculo cada vez que se modifica el valor del control numérico. El ejercicio muestra también la utilidad de poder controlar la escala. En este caso, se puede amplificar una gráfica aparentemente tradicional para notar que al acercarse al origen, ella realmente oscila una infinidad de veces. Es importante considerar un posible error, que es usar un valor de cero para la escala. La escal debe ser un valor mayor que cero, razón por la cual el valor mínimo de la barra que la controla es precisamente 10.

9.5. Control numérico tipo *Botón*

Este tipo de control numérico es el más simple. Consiste en un botón que al ser oprimido hace alguna acción. Muchos de los parámetros del botón son comunes a todos los controles y se pueden consultar dentro de los elementos comunes a los controles.

En la Figura 9.10 se muestra un ejemplo de un control numérico tipo botón en una escena. En la Figura 9.11 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que para este ejemplo el botón se supone se usa para lanzar un cálculo que determinará si la respuesta del usuario es correcta. La respuesta está colocada en otro control numérico (*CamResp*) tipo campo de texto. Observe que, en los cálculos de acción del botón hay una variable *correcto* que de ser correcta la respuesta valdrá 1 y de lo contrario valdrá 2. Puede consultar más a fondo toda esta funcionalidad en el apartado Condicionales y operadores booleanos.

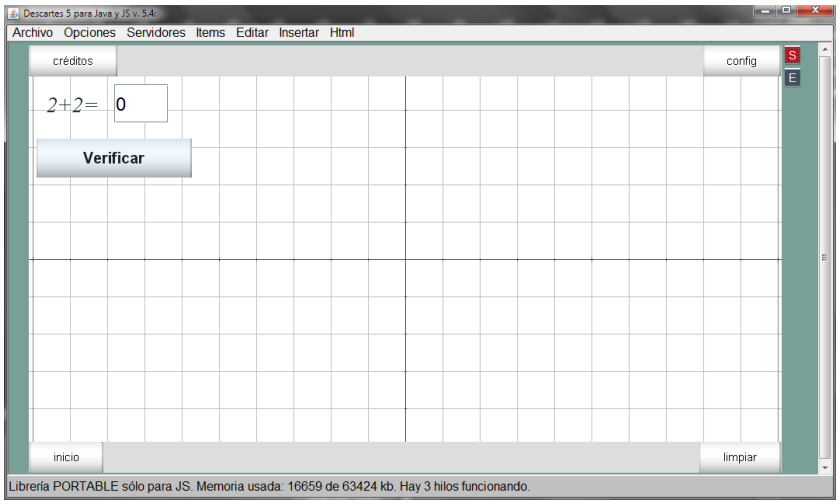

Figura 9.10: Ejemplo del control numérico *Botón*.

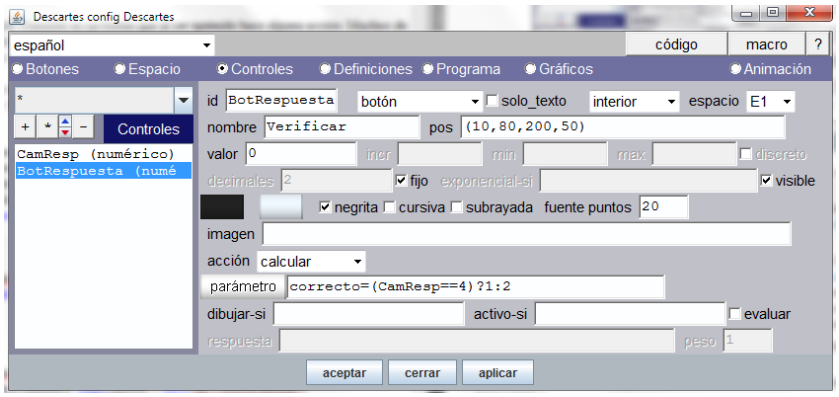

Figura 9.11: Configuración del ejemplo del control numérico *Botón*.

- **botón de selección de color de fuente del botón**: es el botón de la izquierda que por defecto aparece de color negro. Al usarlo, se lanza la herramienta de control de colores, donde se puede seleccionar el color de la fuente del texto del botón.
- **botón de selección de color de fondo del botón**: es el segundo botón que aparece por defecto con un color claro. Al oprimirlo lanza la herramienta de control de colores, donde se puede seleccionar el color del fondo del botón. Como se describe a continuación, es posible asignar una imagen jpg o png al fondo del botón con el campo de texto *imagen*. Si esta imagen tiene transparencia, se recomienda que la transparencia del fondo del botón se ajusta al máximo.
- **negrita**: es un checkbox que debe estar marcado si se desea que el texto del botón aparezca en negritas.
- **cursiva**: es un checkbox que debe estar marcado si se desea que el texto del botón aparezca en letra cursiva.
- **subrayada**: es un checkbox que debe estar marcado si se desea que el texto del botón aparezca subrayado.
- **fuente puntos**: es un campo de texto en el que se introduce el número de puntos que controla el tamaño de la fuente.

- **imagen**: es un campo de texto donde se introduce una ruta relativa a la carpeta donde se encuentra el interactivo, y que apunta hacia a una imagen png o jpg que se ha de usar como fondo del botón. Si la imagen se encuentra en una carpeta, es necesario usar la diagonal sencilla (/) en la ruta. Por ejemplo, si se quisiera usar una imagen *bot.png* en la carpeta *images* que está en la misma carpeta que el interactivo, tendría que escribirse en este campo *images/bot.png*.

Ahora haremos un ejercicio que involucra un botón cuya función es calcular el siguiente valor de una sucesión numérica dados dos números iniciales. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 05. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Controles_05/Controles_05_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio practicamos el uso del control numérico tipo *botón* como un medio para realizar cálculos sucesivos. El valor de la variable *suma* en cada paso resulta seguir una sucesión llamada *Sucesión de Fibonacci*, cuyos primeros valores son 1, 1, 2, 3, 5, 8, 13, 21, 33, ... Como se puede observar, el valor en alguna posición corresponde a la suma de los dos valores anteriores. Se trata de una sucesión recursiva puesto que el valor de alguna posición sólo puede conocerse calculando todos los valores anteriores hasta él. Así pues, el botón realiza el cálculo del valor de la sucesión en la posición en que se encuentra. Algo a notar en este ejercicio es la importancia de inicializar variables en algunas ocasiones. De no inicializar las variables con el valor 1 en este ejercicio, los valores de *suma* nunca cambian. Las variables que no se inicializan siempre adoptan el valor de cero por defecto.

9.6. Control tipo *Gráfico*

Los controles gráficos, a diferencia de los numéricos, consisten en puntos que se pueden manipular directamente con el ratón o, en dispositivos móviles, arrastrarlos en la pantalla táctil. Para agregar un control gráfico se oprime el botón + en el selector *Controles*, con lo que se abre una ventana con un menú adentro. En este menú hay que seleccionar la opción *gráfico* y aceptar para cerrar la ventana. Los identificadores de los controles gráficos empiezan con la letra *g* por defecto.

En la Figura 9.12 se muestra un ejemplo de un control gráfico en una escena. En la Figura 9.13 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que para este ejemplo el control gráfico se usa como un punto que está constreñido a una determinada curva, que se encuentra trazada aparte como un gráfico tipo ecuación. Se muestra la curva sólo con fines de que el usuario pueda ver que el punto sigue dicha trayectoria, aunque no es indispensable mostrarla. Para mayor información sobre el gráfico usado en este caso, consulte el apartado Gráfico ecuación.

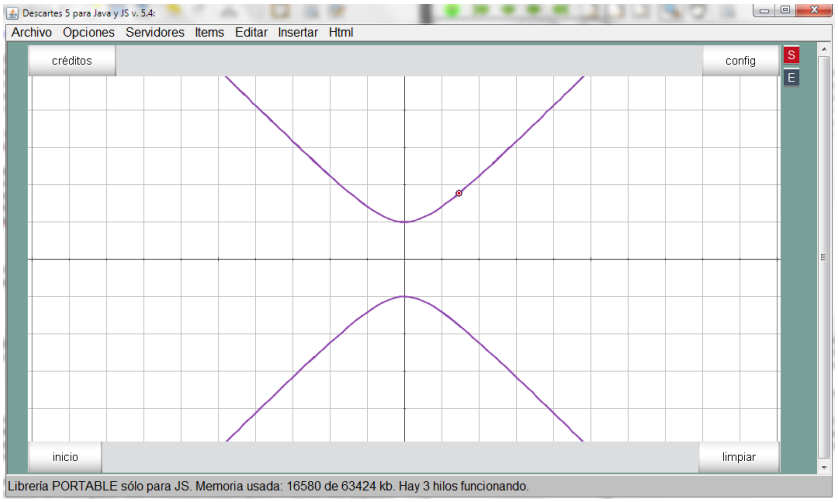

Figura 9.12: Ejemplo del control tipo *Gráfico*.

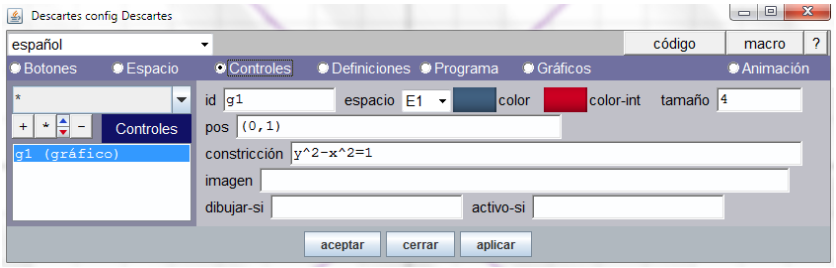

Figura 9.13: Configuración del ejemplo del control tipo *Gráfico*.

- **id**: es un campo de texto donde se agrega el identificador. Este identificador permite, como se verá en breve, extraer propiedades del control como las coordenadas de su posición y si está siendo usado o no. En este sentido no es una variable que guarda un valor como sucede para los identificadores de controles numéricos.
- **espacio**: es un menú donde se escoge en qué espacio aparecerá el control gráfico como el punto interactivo móvil.
- **color**: es un botón que abre la herramienta de control de colores para seleccionar el color del contorno del control gráfico. Como se mencionó, el control gráfico se muestra como un punto o círculo de tamaño ajustable. Es precisamente el color del contorno de este círculo lo que se modifica con este parámetro.
- **color-int**: es otro botón que también abre la herramienta de control de colores para seleccionar el color interior del control gráfico, que es el centro del círculo con que se representa.
- **tamaño**: es un campo de texto en el cual se introduce el número de pixeles que determinan el radio del círculo que representa el control gráfico.
- **pos**: es un campo de texto donde se introduce la coordenada inicial del control gráfico. Por defecto se encuentra en el origen.
- **constricción**: es un campo de texto en el que se introduce una ecuación. Algunas veces es deseable que el control gráfico no pueda moverse a lo largo de todo el espacio que lo contiene, sino que se mueva dentro de una curva permitida. En dicho caso, la ecuación introducida en este

espacio restringirá la posición del control. Por ejemplo, una constricción $x^2 + y^2 = 4$ restringirá al control a una circunferencia centrada en el origen y de radio 2. Note que si la coordenada indicada en *pos* no es parte de la constricción dada al control gráfico, cuando se aplican los cambios, el editor de Descartes puede mostrar el control gráfico en la posición elegida, pero al intentar arrastrarlo éste se ajusta inmediatamente a dicha constricción. En el navegador, el control gráfico siempre aparece en la constricción dada.

- **imagen**: es un campo de texto para la ruta relativa a un archivo de imagen jpg o png que acompañará al control gráfico. Cuando se usa una imagen, la imagen misma puede arrastrarse como si fuera el control gráfico.

En ocasiones es necesario manipular de forma más precisa y/o conocer el valor de la coordenada horizontal y vertical del control gráfico. Hay dos variables asociadas a los controles gráficos que corresponden a estas coordenadas. Son *<identificador del control>.x* y *<identificador del control>.y*. Por ejemplo, la coordenada vertical de un control gráfico con identificador *grf* sería *grf.y*. Estas variables se pueden usar tanto para conocer o imprimir el valor del gráfico, pero también es posible asignarles valores para colocar al gráfico en una posición particular deseada. Más aún, se pueden generar controles numéricos tipo *pulsador* con los nombres de estas variables para poder restringir más finamente el movimiento del control. Por ejemplo, se puede hacer que un control no pueda desplazarse en todo un continuo de valores, sino cada 0.5 unidades, etc.

Hagamos un ejercicio para que todo esto quede más claro. Como motivo de este ejercicio, supongamos que queremos mostrar un histograma del número de personas con edad de 26 y 27 años. El histograma se debe poder construir con columnas cuya altura es ajustable arrastrando un control gráfico. Aprovecharemos el ejercicio para calcular el promedio de las edades.

El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 06. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Controles_06/Controles_06_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Pudimos notar en este interactivo la utilidad de poder controlar cosas mediante arrastre de objetos. Esto se puede extender a la manipulación de figuras geométricas, imágenes, etc. Y son precisamente los controles gráficos los que permiten esta funcionalidad. También observamos la utilidad de colocar instrucciones o asignaciones en el algoritmo *CÁLCULOS*. No obstante, como se menciona en el apartado de dicho algoritmo, hay que usarlo con ciertas reservas ya que puede acabar innecesariamente gastando recursos de cálculos en repetir cálculos innecesarios. Finalmente, pudimos notar que en un ejercicio tan sencillo como calcular el promedio de algún dato se presentó un error. Aunque se puede argumentar que el no tener individuos no tiene sentido para un promedio, el interactivo se podría pulir un poco para que no generase errores en la ventana de comandos, y para que el texto mostrado al usuario fuera un poco más explicativo. Algo así como *No se puede buscar un promedio si no se tiene elementos*. Las estrategias para lograr este tipo de comportamiento se consideran en otros ejercicios.

Hagamos otro interactivo con controles gráficos que permite además una herramienta para el programador muy útil. Ya se sabe que hay dos tipos de coordenadas: las absolutas y las relativas. Algunos gráficos permiten el uso de ambas; otros como el texto sólo manejan las absolutas; y todos los controles manejan únicamente las absolutas. Por otra parte, los controles gráficos se manejan en coordenadas relativas. En ocasiones se requiere conocer las coordenadas absolutas de un punto en un espacio para, por ejemplo, saber qué coordenadas darle a un control o un texto. Aunque el control gráfico maneja coordenadas relativas, podemos convertirlas a absolutas y así, de manera ágil, moverlo a alguna posición y nos dirá inmediatamente tanto las coordenadas relativas como las absolutas de la misma. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 07. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/

Controles_07/Controles_07_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

El resultado de este ejercicio es un control gráfico que puede desplazarse a cualquier parte del espacio y nos indica las coordenadas relativas así como las absolutas. Si al programador le gusta un punto particular para colocar digamos un botón, puede colocar el control gráfico ahí y conocerá las coordenadas absolutas que debe usar para el botón.

Este tipo de herramientas son útiles para el programador. Una vez colocados los objetos en las posiciones deseadas, el control gráfico y el punto con su texto se vuelven innecesarios y es conveniente que no se muestren. No es necesario eliminarlos completamente, por lo que se pueden solo ocultar aprovechando los parámetro *dibujar-si* tanto del control como del punto. De hecho, si eventualmente ha de cambiarse algo en el interactivo, es posible que se requiera usarlos otra vez, razón por la cual conviene no eliminarlos.

A lo largo de este ejercicio se usaron variables del tipo *E1._w*, *E1._h*, *E1.Ox*, *E1.Oy* y *E1.escala*. Todas estas variables son intrínsecas a Descartes. Si se desea estudiarlas más a fondo puede consultar el apartado sobre variables de espacio.

9.7. Control tipo *Texto*

Los controles tipo *texto* consisten en bloques de texto enriquecido que contienen un marco que los delimita. Es posible editar el texto dentro del recuadro en el interactivo, y no necesariamente dentro del editor de configuraciones. Adicionalmente, se puede mostrar un par de textos tipo pregunta y respuesta mediante un botón que se incluye en la esquina inferior derecha dentro del recuadro de texto.

En la Figura 9.14 se muestra un ejemplo de un control tipo texto en una escena. En la Figura 9.15 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

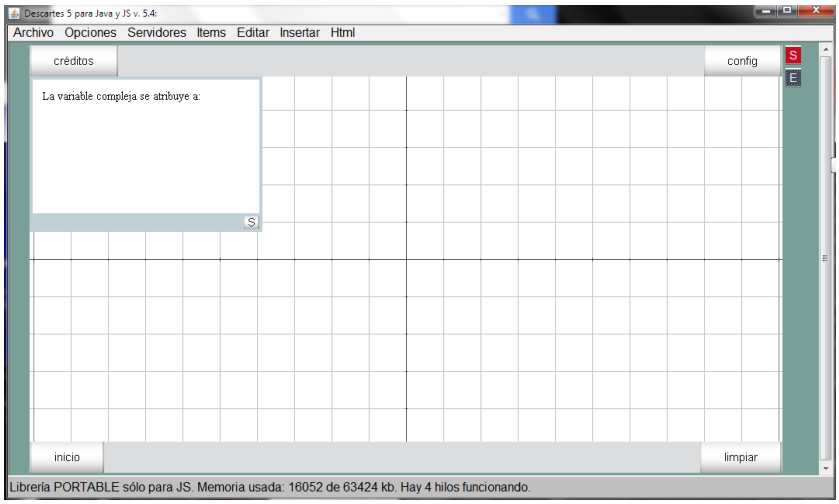

Figura 9.14: Ejemplo del control numérico *Texto*.

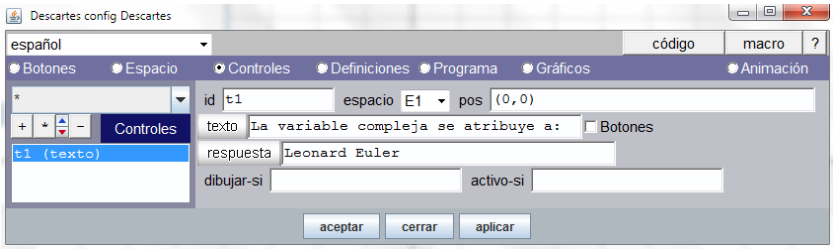

Figura 9.15: Configuración del ejemplo del control tipo *Texto*.

- **texto**: es un campo de texto acompañado de su botón *texto*. Se puede introducir el texto directamente en el campo u oprimir el botón para mostrar la ventana de introducción de texto enriquecido. Si requiere información detallada sobre la forma de introducción de texto, puede consultar el apartado sobre la herramienta de introducción de textos. La idea de uso de este texto es que el programador incluya una pregunta aquí que el usuario ha de responder. Por ejemplo: *Indica en qué año vivimos:*
- **botones**: la funcionalidad de este botón no es soportada en el intérprete actual de Descartes, por lo que puede ignorarse.
- **respuesta**: es otro campo de texto acompañado del botón *respuesta*. Se puede introducir el texto directamente en el campo u oprimir el botón para desplegar la ventana de texto enriquecido. Consiste en la respuesta a ser mostrada como correcta cuando el usuario quiera corroborar la suya. Note que esta respuesta sólo es mostrada. No se hace una comparación para determinar si lo que el usuario introdujo es correcto o no. Un ejemplo de la respuesta podría ser *El año en que vivimos es 2016*.

El interactivo mostrará el texto introducido en *texto*. El usuario podrá poner en ese mismo recuadro su respuesta a la pregunta. Si se incluyó algo en el campo *respuesta* del control, aparece en la esquina inferior de la caja de texto en el interactivo un botón con la letra *S*. Si el usuario oprime dicho botón, se muestra lo que se haya introducido en el campo *respuesta* y aparece ahora un botón *T* con el que se regresa a la pregunta. La funcionalidad del control tipo texto prácticamente no se usa dado que no permite una comparación de respuesta; sólo muestra la respuesta correcta.

9.8. Control tipo *Audio*

Este control consiste en un reproductor de audio que puede ser activado en el interactivo. Se puede colocar en el exterior, interior, y básicamente en cualquier lugar como si fuera un control numérico. Reproduce archivos mp3 y wav.

En la Figura 9.16 se muestra un ejemplo de un control tipo audio en una escena. Esta escena está visualizada desde el navegador, y no desde el editor de Descartes. Ello debido a que el editor de Descartes no muestra el control tipo audio con su funcionalidad y esto sólo se visualiza cuando está cargado desde un navegador. En la Figura 9.17 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

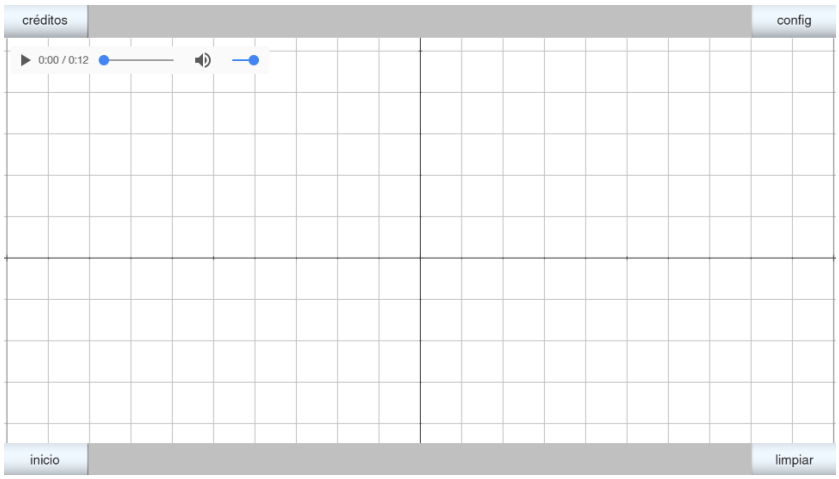

Figura 9.16: Ejemplo del control tipo *Audio*.

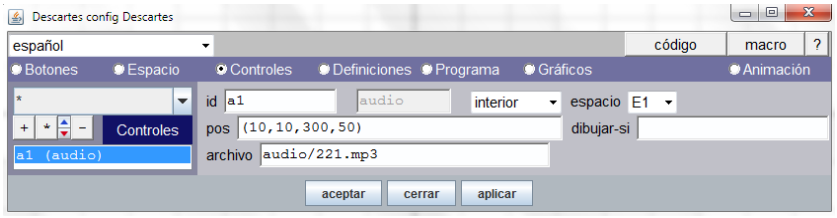

Figura 9.17: Configuración del ejemplo del control tipo *Audio*.

- **archivo**: es un campo de texto en el que se introduce una ruta relativa a donde se encuentra el interactivo y que apunta al archivo de audio a reproducir. Recuerde que si el archivo se encuentra en una subcarpeta, es necesario usar la diagonal sencilla para separar carpetas. Por ejemplo, *audio/ruido.mp3*.

Existen algunas funciones y variables intrínsecas que comienzan con el nombre del identificador del control de audio (para información más detallada se puede consultar el apartado sobre funciones de controles de audio y video y variables de controles de audio y video). Las funciones llevan un paréntesis de apertura y cierre. Cuando la función no maneja argumentos, el interior del paréntesis está vacío. Por ejemplo, <*identificador del control de audio*>.*play*() con la que se puede controlar la reproducción del archivo. La única variable de audio y video no lleva paréntesis.

Cabe mencionar que el editor de Descartes no permite la reproducción de los archivos directamente ahí. Así que para ver los controles en el interactivo y poder reproducir los archivos es necesario guardar y abrir el interactivo en el navegador. Es decir, solamente aplicar los cambios en este caso no basta.

Hagamos un breve ejercicio en donde se reproducen un par de frecuencias cuando se eligen a partir de un menú. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Controles 08. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Controles_08/Controles_08_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*. Los archivos de audio para este ejercicio se encuentran en 221.mp3 y 371.mp3, y también están dentro del archivo *DocumentacionDescartes5.zip*.

Algo importante a notar de este ejercicio es que el reproductor que se visualiza en el navegador es nativo al navegador en cuestión y no a Descartes. Puede no verse igual en Chrome que en Firefox,

debido a cambios que se hacen en las versiones de dichos programas. Pero los botones básicos del control suelen ser los mismos.

También es importante tener en mente que los controles de audio y video y la reproducción de archivos no es posible en el Editor de Descartes. Es necesario cargar el interactivo en un navegador. Es por ello que en estos pasos se indica explícitamente que hay que aplicar, guardar y cargar el interactivo en un navegador.

Adicionalmente, fue necesario usar una animación sólo para que el tiempo de reproducción de uno de los archivos se despliegue. Si no se quisiera desplegar, no sería necesario tener la animación. Si se requiere más detalles sobre las animaciones, se puede consultar el apartado Animación.

9.9. Control tipo *Video*

Este control consiste en un reproductor de video que puede ser activado en el interactivo. Su funcionamiento es muy similar al control de audio. Se utiliza para reproducir archivos mp4.

En la Figura 9.18 se muestra un ejemplo de un control tipo video en una escena. Esta escena está visualizada desde el navegador, y no desde el editor de Descartes. Ello debido a que el editor de Descartes no muestra el control tipo video con su funcionalidad y esto sólo se visualiza cuando está cargado desde un navegador. En la Figura 9.19 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

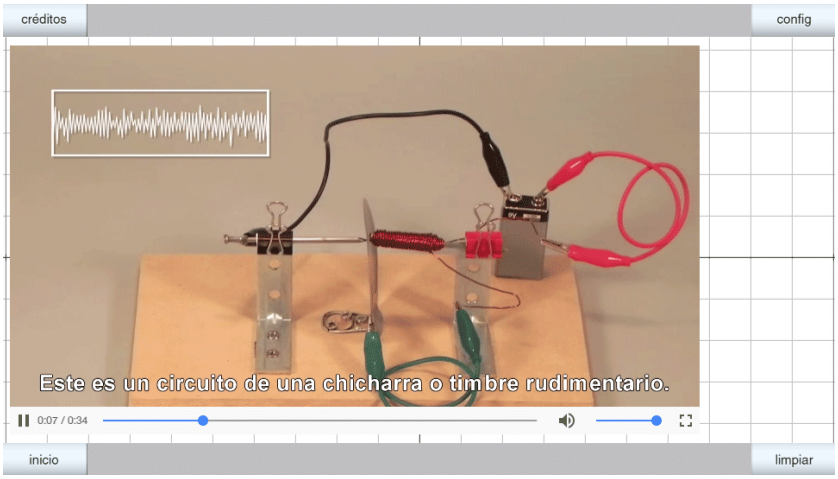

Figura 9.18: Ejemplo del control tipo *Video*.

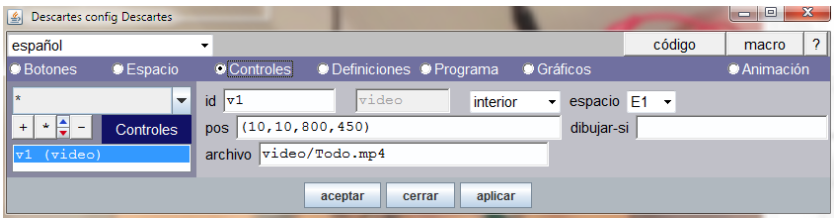

Figura 9.19: Configuración del ejemplo del control tipo *Video*.

- **archivo**: es un campo de texto en el que se introduce una ruta relativa a donde se encuentra el interactivo y que apunta al archivo de video a reproducir. Recuerde que si el archivo se

encuentra en una subcarpeta, es necesario usar la diagonal sencilla para separar carpetas. Por ejemplo, *video/peli.mp4*.

Los controles de video en Descartes cuentan con las funciones intrínsecas: <identificador del control>.play(), <identificador del control>.stop(), <identificador del control>.pause(), <identificador del control>.currentTime(<segundo de inicio del video>). Adicionalmente, también cuentan con la variable intrínseca <identificador del control>.currentTime que guarda el valor del tiempo en que se encuentra el video en segundos. Como se observa, las funciones y variables intrínsecas son exactamente las mismas que las descritas para controles de audio, que se pueden consultar más detalladamente en el apartado sobre funciones de controles de audio y video y variables de controles de audio y video.

Al igual que para los controles de audio, el control de video que aparece en un navegador puede diferir de otro en funcionalidad. Algunos pueden desplegar, por ejemplo, la opción de extender el video a pantalla completa, otros no. Todo esto depende del reproductor de video del navegador en que se muestran los interactivos, y no de Descartes.

Se recomienda que cuando se use un control de video, el parámetro *pos* incluya las cuatro entradas de posición en la horizontal, en la vertical, ancho y alto. Conviene que el ancho y alto del control correspondan al ancho y alto del video, o mínimo que respeten la misma proporción.

Si se llegara a requerir una portada para un video (una imagen estática que se muestra al inicio del video), esto se logra guardando en la misma carpeta que el video una imagen png cuyo nombre es el mismo que el del video y cuyas dimensiones también son iguales a las del video.

Debido a la similitud entre la funcionalidad del control de audio y video, no se incluye un ejercicio para este apartado. Se recomienda, sin embargo, que el usuario haga experimentos con algún video.

9.10. Elementos comunes a todos los controles

A continuación se muestran los elementos comunes a los controles.

- **id**: es un campo de texto conocido como *identificador*. Es el "nombre" que se le asigna al control dentro de Descartes, y no el que se observa dentro del interactivo. Cuando se quiere hacer referencia al valor del pulsador como una variable, éste es el nombre de la variable que lo representa. Como finalmente es una variable, cabe mencionar que ningún identificador puede empezar con un número. El identificador se puede definir en la ventana que aparece al pulsar el botón de agregar un control nuevo (arriba del panel de la lista de controles y que tiene un signo + en su interior). No obstante, posteriormente siempre se puede cambiar en el campo de texto *id*.
- **menú del tipo de control**: es un menú que por defecto aparece en la opción *pulsador*. Determina el tipo de control que se desea. Sus opciones son *pulsador*, *campo de texto*, *menú*, *barra* y *botón*. Los detalles de cada uno de estos tipos de control se detallan a lo largo del apartado de *Controles*.
- **menú de localización del control**: es un menú que por defecto aparece en la opción *sur*. Determina donde ha de colocarse el control.

  sur: hace que el control se coloque en una barra horizontal insertada en la parte inferior del interactivo, fuera de los espacios. Pueden incluirse varios controles con esta opción, y todos los que la tengan aparecerán en dicha barra.

  norte: hace que el control se coloque en una barra horizontal insertada en la parte superior del interactivo, fuera de los espacios. Pueden incluirse varios controles con esta opción, y todos los que la tengan aparecerán en dicha barra.

  este: hace que el control se coloque en una barra vertical insertada en la parte derecha del interactivo, fuera de los espacios. Pueden incluirse varios controles con esta opción, y todos los que la tengan aparecerán en dicha barra.

oeste: hace que el control se coloque en una barra vertical insertada en la parte izquierda del interactivo, fuera de los espacios. Pueden incluirse varios controles con esta opción, y todos los que la tengan aparecerán en dicha barra.

exterior: hace que el control esté fuera del interactivo, a diferencia de las anteriores cuatro opciones. Un control en el *exterior* se puede mostrar en una ventana que aparece al hacer clic derecho sobre el interactivo. Los controles colocados en el exterior suelen usarse con propósitos internos para el programados. Es decir, no suelen ser parte del interactivo final.

interior: hace que el control esté dentro del interactivo, y dentro de un espacio en particular. En este caso, el campo de texto *pos* explicado en breve determina las coordenadas y tamaño del control en pixeles dentro del espacio que lo contiene.

escenario: se relaciona con controles insertados en el flujo de texto de un *Discurso*. Los controles ahí insertados tendrán *escenario* como opción del menú de localización, ya que no estarán en un espacio particular ni en el exterior. En la parte de Discursos se detalla la funcionalidad de los discursos.

- **espacio**: es un menú dentro del cual se determina cuál de los espacios existentes alojará al control en cuestión. Sólo tiene sentido cuando el menú de localización del control está puesto en *interior*.
- **nombre**: es un campo de texto en el cual se introduce el nombre del pulsador como habrá de verse en el interactivo. Sólo aplica para controles de tipo numérico. A diferencia del *identificador*, este nombre no es la variable con la que se relaciona al control, sino simplemente el nombre que el usario final verá en el interactivo. Aunque por defecto el nombre del control se copia del identificador, es posbile, y muchas veces deseable, que el nombre se cambie por otro. Por ejemplo, un control numérico de tipo pulsador cuyo objeto es ajustar una distancia en un interactivo puede tener como identificador *distPl*. Así, el programador rápidamente lo puede localizar como el pulsador que controla la distancia. Sin embargo, es preferible que este identificador sea su nombre final en el interactivo. En su lugar, el nombre final podría ser algo como *control de distancia*. En caso que no se desee mostrar explícitamente el nombre del pulsador, se puede introducir el texto *_nada_* en este campo.
- **pos**: es un campo de texto que determina las coordenadas del control. A diferencia de los gráficos, los controles siempre están determinados en coordenadas absolutas. Así pues, si *pos* tiene *(150,30)*, la esquina superior izquierda del control estará 150 pixeles a la derecha de la esquina superior izquierda del interactivo y 30 hacia abajo. Las dimensiones del control también se pueden determinar en *pos* si se incluyen cuatro coordenadas en lugar de dos. En este caso, las dos últimas coordenadas serán el ancho y alto, en pixeles, del control. Usando el ejemplo anterior, si el control tiene *(150,30,200,45)* en *pos*, entonces estará en el mismo lugar, pero además tendrá un ancho de 200 pixeles y un alto de 45.
- **valor**: es un campo de texto que indica el valor inicial que tendrá el control en el interactivo. Así pues, es un número en la mayoría de las ocasiones, aunque puede ser también un texto en algunos casos. Su funcionalidad es un poco distinta para el caso de los menús, y se detalla en la descripción de dicho tipo de control numérico.
- **incr**: es un campo de texto que indica el tamaño del incremento de un control numérico. Aplica particularmente para el pulsador y la barra. Por ejemplo, si se desea que un pulsador aumente de 0.5 en 0.5 unidades, se colocaría 0.5 en este campo de texto.
- **min**: es un campo de texto que indica el valor mínimo que el control puede adoptar. Si se deja vacío quiere decir que no hay un límite inferior para el parámetro.
- **max**: es un campo de texto que indica el valor máximo que el control puede adoptar. Si se deja vacío quiere decir que no hay un límite superior para el parámetro.
- **discreto**: es un checkbox que cuando está marcado obliga al parámetro a adoptar valores que difieren de su valor inicial sólo en múltiplos del incremento indicado. Por ejemplo, un pulsador

con valor inicial 0.05, valor mínimo de 0, incremento de 0.1 y no considerado discreto variará su valor de 0.05 a 0.00 cuando se hace clic para disminuir su valor un paso. Sin embargo, uno igual pero con el checkbox *discreto* marcado se mantendrá en 0.05 pues 0.00 no difiere en un múltiplo del incremento.

- **decimales**: es un campo de texto en el que se indica cuántos decimales se han de mostrar para el parámetro. Tras evaluar el parámetro, se redondeará el valor a ese número de decimales y ése es el valor que se mostrará.
- **fijo**: es un checkbox que determina si el número de decimales elegido ha de mostrarse siempre (aún cuando los decimales en cuestión no sean significativos) cuando está marcado, o bien si se ha de usar la notación ajustada en la que sólo se muestran el número de decimales indicado si son significativos cuando está desmarcado.
- **exponencial-si**: es un campo de texto que admite una expresión booleana (es decir, cuyo valor puede ser 0 ó 1). Si el valor de la expresión es 0, nunca se usará la notación exponencial, pero si es 1, es posible que ésta se muestre si el interactivo lo determina pertinente.
- **visible**: es un checkbox que cuando está marcado hace visible el valor del parámetro y cuando no lo mantiene invisible.
- **acción**: es un menú que determina qué hará el interactivo cada vez que es usado. Las opciones disponibles en dicho menú son:

  calcular: hará los cálculos indicados en el *parámetro*, que se explica adelante. Pueden consistir en asignaciones de valor a variables y llamados a funciones.

  inicio: carga el interactivo como si fuera la primera vez que se abre, ignorando cambios que el usuario haya hecho sobre el mismo posteriormente. Su función es la misma que la del botón *inicio* que aparece en la esquina inferior izquierda en un interactivo recién creado.

  limpiar: cuando hay gráficos que estén definidos de tal forma que dejen rastro, esta opción permite que cada vez que el control en cuestión es usado, dichos rastros se limpien. Su función es la misma que la del botón *limpiar* que aparece en la esquina inferior derecha en un interactivo recién creado.

  animar: lanza la animación definida en el selector *Animación*.

  abrir URL: permite abrir una dirección URL al hacer uso del control. La dirección URL destino se especifica en *parámetro* como se explica más adelante. Esta funcionalidad sólo puede usarse en el interactivo visualizado en un navegador. De intentarse esto dentro del editor, no funcionará.

  abrir escena: permite abrir una escena que se encuentra en la misma carpeta que el interactivo. La escena se especifíca en *parámetro*, cuya descripción se aborda adelante. Al igual que para *abrir URL*, esta funcionalidad sólo está disponible cuando se usa el interactivo en el navegador. Es decir, no está disponible desde el editor.

  reproducir: permite reproducir un archivo de audio tipo *.mp3*. Si el archivo se encuentra en la misma carpeta que el interactivo basta introducir su nombre en el campo de texto *parámetro*. También es posible incluir en *parámetro* la ruta a una subcarpeta en caso que el archivo de audio se encuentre en ella. Cada vez que el valor del control es modificado, el archivo de audio se reproducirá. Esta funcionalidad es sólo válida ya que se visualiza el interactivo en un navegador y no en el editor.
- **parámetro**: Es un botón acompañado de un campo de texto. El botón abre una ventana en la cual el parámetro en cuestión se puede introducir cómodamente como texto. El campo de texto tiene la misma función, pero en una sola línea.

  Si la acción del control es *calcular*, el parámetro consiste en asignaciones y/o llamados a funciones.

Si la acción es *abrir URL*, el parámetro puede llevar una dirección relativa a la carpeta en que se encuentra el interactivo o una dirección absoluta comenzando por ejemplo con *http://*.

Si la acción es *abrir Escena*, el parámetro ha de llevar el nombre de la escena a abrir con todo y su extensión *.html* que se encuentra en la misma carpeta o en una subcarpeta respecto a donde se encuentra el interactivo que la llama.

- **dibujar-si**: es un campo de texto que admite una expresión booleana. De valer 1 dicha expresión, el control en cuestión estará visible, mientras que de valer 0 no lo estará.
- **activo-si**: es un campo de texto que admite una expresión booleana. De valer 1 dicha expresión, el control en cuestión estará activo, mientras que de valer 0 no lo estará. A diferencia de *visible*, *activo* determina si el control puede usarse o no. El control puede estar visible, pero si está inactivo se ve de un color más opaco y el usuario no podrá interactuar con el control.
- **evaluar**: es un checkbox que cuando está marcado permite que el control sea usado con fines de evaluación. Esta funcionalidad, y la del parámetro *respuesta* sólo son válidas para el control numérico tipo *campo de texto*.
- **respuesta**: es un campo de texto donde se introduce la posible respuesta contra lo que se comparará lo que el usuario haya entrado. Existen varias opciones de evaluación:

  solo texto: Cuando el control es de tipo sólo texto, se compara el texto con lo que el usuario introduce. Acepta los comodines típicos de manejo de texto: asterisco (*) para indicar que la palabra puede llevar texto libre de cualquier longitud, siempre y cuando contenga la palabra, interrogación (?) para indicar que esa posición se puede sustituir por cualquier letra, flanquear el texto entre comillas sencillas (‘) para indicar que se ignoren la diferencia entre mayúsculas y minúsculas en la comparación, y flanquearlo entre acento grave (`) y el agudo o sencillo para que se ignoren acentos y la diferencia entre *n* y *ñ* en la comparación. Si varias distintas opciones son consideradas como correctas, se separan con el caracter *pipe* (|).

  Cuando el control acepta números, los correctos se deben introducir como intervalos numéricos de la forma (a, b), $[a, b)$, $(a, b]$ y $[a, b]$. Los paréntesis representan intervalos abiertos (se consideran correctos numeros desde o hasta aquél indicado en el interavlo, pero sin incluirlo como opción correcta) y los corchetes cuadrados representan intervalos cerrados (el número al lado del corcehte sí se considera también como opción correcta). Si la respuesta es sólo un número, aún así debe introducirse como intervalo. Por ejemplo si la respuesta es 3, debe introducirse como [3,3].

Cabe mencionar que las opciones del menú *acción* se ejecutan cuando el control es usado de alguna manera. Esto no sólo implica hacer uso directo del control. Por ejemplo, un pulsador puede modificarse con las flechas para subir o bajar su valor. No obstante, también se puede introducir texto en el campo de texto que por defecto acompaña a los pulsadores y oprimir la tecla INTRO. Esto también lanzará la acción asignada a ese control.

Cuando se considera la opción *evaluar*, aparece también una opción *peso*. Sin embargo, esta opción ya no se puede usar debido a que el intérprete de Descartes ya no la toma en cuenta. Para saber si lo que introdujo el usuario es correcto se incluye la variable *<identificador del control>.ok*. Dicha variable vale 1 en caso de una respuesta correcta y 0 en caso de una incorrecta.

Abordemos un ejercicio para practicar un poco lo aprendido en esta parte. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en ComunesControles. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/ComunesControles/ComunesControles_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio se observaron algunas de las funciones más típicas de los parámetros comunes a la mayoría de los controles. En este caso, el control *grosBr* se mantuvo fuera del interactivo puesto

que el grosor es algo que quedará fijo en el interactivo. Es decir, el programador querrá intentar varios grosores y ver cuál queda mejor al final del interactivo, pero el control como tal no deberá quedar disponible para el usuario final.

Capítulo 10

El selector *Programa*

El selector *Programa* incluye ya parte de la programación dura que se hace con Descartes. Consta básicamente de dos tipos de algoritmos presentes por defecto en cualquier escena (*INICIO* y *CÁLCULOS*), y de un elemento conocido como evento.

El selector *Programa* contiene un panel a la izquierda igual al de los otros selectores. Es decir, hay un botón + con el cual se puede agregar un elemento nuevo. El único elemento nuevo que puede agregarse es, de hecho, el elemento tipo evento. Los algoritmos *INICIO* y *CÁLCULOS* se encuentran presentes por defecto. El panel muestra la lista de elementos del selector *Programa*. El botón *Programa* muestra la lista de elementos en forma de texto (para una edición más ágil) al igual que sucede en otros selectores. A continuación se describen los componentes del selector *Programa* con más detalle:

10.1. INICIO

El algoritmo Inicio permite hacer los cálculos iniciales que dejan listo al interactivo para ser usado. Estos cálculos pueden incluir asignaciones condicionales y llamados a funciones. En la Figura 10.1 se muestran los componentes del algoritmo *Inicio*.

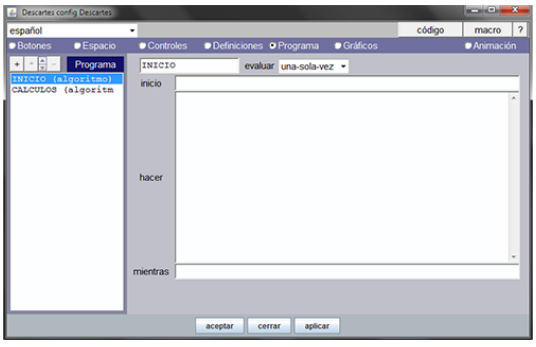

Figura 10.1: Visualización del algoritmo *Inicio*.

- **identificador**: es un campo de texto para el nombre del algoritmo. Por defecto trae el nombre *INICIO*, y es recomendable no cambiarlo.
- **evaluar**: es un menú mediante el cual se indica cuándo han de evaluarse las instrucciones del algoritmo:

  una-sola-vez: Sólo se evalúa el algoritmo en cuestión una sola vez al inicio de la escena. Como en este caso se trata del algoritmo *INICIO*, este menú se encuentra por defecto en *una-sola-vez*.

Si un algoritmo sólo se ha de realizar una vez, es conveniente dejar esta opción marcada ya que reduce el número de cálculos y evita que el interactivo se ralentice.

siempre: El algoritmo se evalúa siempre que el usuario haga un cambio en el interactivo, tal como modificar un control. Esta opción se usa para algoritmos que es necesario efectuar constantemente. Es preciso tener cuidado de hacer estos algoritmos cuan pequeños sea posible ya que, al repetirse constantemente, pueden ralentizar el interactivo.

- **inicio**: es un campo de texto donde se pueden incluir también asignaciones, llamados a funciones, etc. que se hacen al inicio del algoritmo. Es posible que un algoritmo requiera repetirse un cierto número de veces y que en dichas repeticiones algunas variable tomen distintos valores. En este campo de texto se incluyen las asignaciones, etc. que se hacen antes de realizar las repeticiones. De ahí que su nombre sea *inicio*. Cuando se hacen diversas instrucciones en este campo de texto, se debe usar el punto y coma (;) para separar cada una.
- **hacer**: es un panel en que se puede introducir las diversas instrucciones. Si el algoritmo involucra hacer instrucciones de forma cíclica, estas instrucciones deben ir precisamente en este panel. Si al algoritmo no involucra instrucciones de forma cíclica, da lo mismo si las instrucciones se introducen en este panel o si se introducen en el campo de texto *inicio*.
- **mientras**: es un campo de texto en el cual se introduce una condicional que, de ser cierta, obligará que se repitan las instrucciones en *hacer* hasta que la condicional deje de ser cierta. Si no se incluye texto en este campo, Descartes considera que las instrucciones en *hacer* sólo se harán una vez.

10.2. CÁLCULOS

El algoritmo Cálculos puede incluir instrucciones, asignaciones, condicionales y llamados a funciones. Dicho algoritmo se llama cada vez que el usuario hace un cambio en el interactivo. Su disposición en el Editor de configuraciones es idéntica al del algoritmo INICIO, con la salvedad de que el menú *evaluar* se encuentra por defecto en *siempre*. Aunque es posible cambiar dicho menú, se recomienda que se mantenga en *una-sola-vez* para el algoritmo *INICIO* y en *siempre* para el algoritmo *CÁLCULOS*.

Abordemos un ejercicio para practicar los algoritmos INICIO y CÁLCULOS. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Algoritmos 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Algoritmos_01/Algoritmos_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio se puede notar que el algoritmo INICIO se usa para asignaciones o instrucciones que no se deben actualizar todo el tiempo. Si se desea que se actualicen todo el tiempo, conviene usar el algoritmo CÁLCULOS. Pero hay que evitar saturar dicho algoritmo con cosas que no necesariamente deben calcularse de forma constante puesto que puede ralentizar el interactivo.

Note además que las asignaciones en este ejercicio se hacen en el panel *hacer* de los algoritmos INICIO y CÁLCULOS. Bien se pueden colocar en el panel *inicio*, dado que no se están usando los algoritmos como ciclos (el campo *mientras* está vacío, que implica que lo que se incluya en *hacer* sólo se hará una vez y no se repetirá cíclicamente).

10.3. Eventos

Un evento es una acción, o conjunto de acciones que se realizan cuando una cierta condición se cumple. Es posible determinar la frecuencia con que se implementarán las acciones, como se describe a continuación. Las acciones disponibles son las mismas que se ejecutan mediante el uso de controles. En la Figura 10.2 se muestran los componentes del algoritmo *evento*.

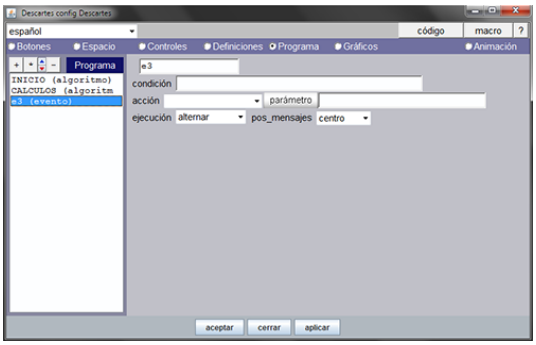

Figura 10.2: Visualización del algoritmo *Evento*.

- **identificador**: es un campo de texto en el cual se introduce el identificador del evento. Este identificador suele servir sólo para que el programador localice el evento en cuestión. No suele hacerse referencia al mismo en otras partes del programa.
- **condición**: es un campo de texto en el cual se introduce la condición que ha de cumplirse para que el evento sea ejecutado. Si la condición es verdadera, el evento se lanzará.
- **acción**: es un menú en el que se elige la acción que lanzará el evento en caso de cumplirse su condición. Este menú es el mismo que el menú *acción* en los controles y sus opciones se detallan en dicha sección, de tal forma que cualquiera de las acciones posibles por los controles también será posible mediante un evento.
- **parámetro**: es un campo de texto en el que se incluyen las instrucciones que se han de ejecutar en caso que la acción del evento sea *calcular*; o la dirección URL en caso que la acción sea *abrir URL*; o la dirección de la escena si la acción es *abrir Escena*; o la ruta al archivo a reproducir si la acción es *reproducir*. Funciona igual que para las acciones de los controles numéricos. Incluye un botón que lanza una ventana de edición de texto para una más cómoda introducción del parámetro.
- **ejecución**: es un menú mediante el cual se determina, dependiendo de la condición del evento, exactamente cuándo ha de ejecutarse la acción. Sus opciones son:

  alternar: hace que la acción se ejecute cada vez que la condición pasa de ser falsa a verdadera.

  una-sola-vez: hace que la acción se ejecute una única vez, que es la primera vez que la condición pasa de ser falsa a verdadera.

  siempre: hace que la acción se ejecute siempre que la condición sea verdadera, y no sólo cuando pasa de ser falsa a verdadera, como en el caso de *alternar*.
- **pos_mensajes**: es un menú que actualmente ya no funciona con el nuevo interprete, por lo que puede ser ignorado.

Hagamos un ejercicio para practicar los eventos y, en particular, las diferencias en sus modos de ejecución. Se busca un interactivo que haga que una imagen se pegue al mouse siempre que este esté oprimido. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Algoritmos 02. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Algoritmos_02/Algoritmos_02_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio se puede ver que una condición por sí sola no siempre especifica qué debe hacer un evento. El parámetro *ejecución* del mismo determina de forma más precisa su comportamiento.

En este interactivo se usaron variables intrínsecas de Descartes relacionadas a acciones del mouse. Éstas se pueden consultar en el apartado sobre variables del mouse. Igualmente, se puede consultar el apartado sobre condicionales y operadores booleanos dado que también se usaron condicionales en el evento mismo.

Los eventos resultan particularmente útiles cuando se desea que un botón controle una animación además de ejecutar otras instrucciones que no se quiera incluir dentro de la animación misma. En ese caso, la variable que controla la animación se puede asociar a un evento cuya accion es lanzar o detener la animación.

Capítulo 11

El selector *Definiciones*

Las definiciones también incluyen gran parte de la programación dura que se hace en Descartes. Éstas consisten en variables, cuyo objeto es iniciar una variable tempranamente en el programa; vectores, que pueden verse como una variable que puede tener simultáneamente una hilera de valores dentro de ella; matrices, que es una extensión de un vector más allá de una hilera de valores; y funciones, que suelen ser un conjunto de instrucciones que se puede elegir cuándo se lanzan y además tienen la capacidad de realizarse cíclicamente.

En adelante veremos con más detalle en qué consiste y cómo funciona cada una de las definiciones. También podremos ver cómo éstas permiten manejar los datos de una forma más cómoda, simplificada y ágil y, a su vez, permiten reducir el código de un interactivo en gran medida.

11.1. Variable

La variable permite usar una variable a la que se le asigna un valor o expresión desde el inicio en que se carga el interactivo. En sí, cuando recibe una expresión se comporta de forma muy similar a la definición tipo función (que se puede consultar en el apartado Función), por lo que ya no se utiliza mucho.

En la Figura 11.1 se muestra un ejemplo del uso de una definición tipo variable en una escena. En la Figura 11.2 se muestra la configuración del editor de configuraciones para lograr dicho ejemplo.

Note que la asignación a la variable no es un valor como tal, sino una expresión que en este caso da la distancia de un punto al origen. En la Figura 11.1 se muestra un texto en el que se imprime el valor de dicha variable. Claro que previamente debieron definirse los valores de x e y (por ejemplo, en el algoritmo INICIO para que el cálculo genere el resultado correcto.

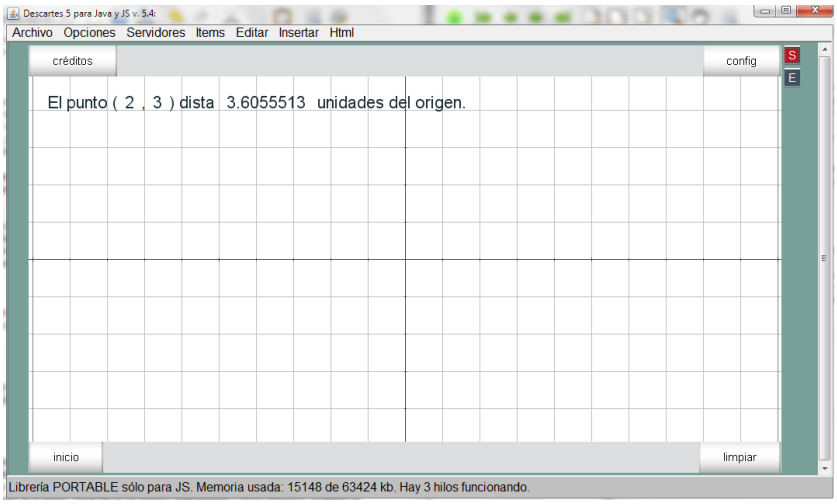

Figura 11.1: Ejemplo del la definición tipo *Variable*.

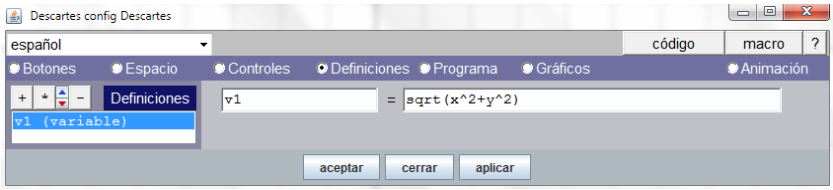

Figura 11.2: Configuración del ejemplo de la definición tipo *Variable*.

- **identificador**: es un campo de texto en el que se introduce el nombre del identificador asociado a la variable en cuestión. Este identificador es el que se usa cada vez que se necesite usar el valor de la misma.
- **valor de la variable**: es un campo de texto que se encuentra a la derecha del signo igual de la variable y en el que se introduce el valor de la misma. Puede llevar un número explícitamente o hacer referencia, mediante su identificador, a otra variable.

Hagamos un muy breve ejercicio para practicar este tipo de definición. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Definiciones 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Definiciones_01/Definiciones_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Notamos en este ejercicio que las variables pueden recibir una expresión, en cuyo caso se realizarán los cálculos de la expresión y el valor se asignará a la variable.

11.2. Función

Las funciones involucran una o un conjunto de instrucciones que se pueden activar sólo en ciertas ocasiones. Tienen la virtud de que se pueden agrupar ciertas instrucciones que se repiten muchas veces a lo largo de un programa en un solo bloque de código. Es probablemente el núcleo sobre el cual gira la programación en general, por lo que se le dará un énfasis particular en esta documentación. En la Figura 11.3 se muestran los elementos de este tipo de definición.

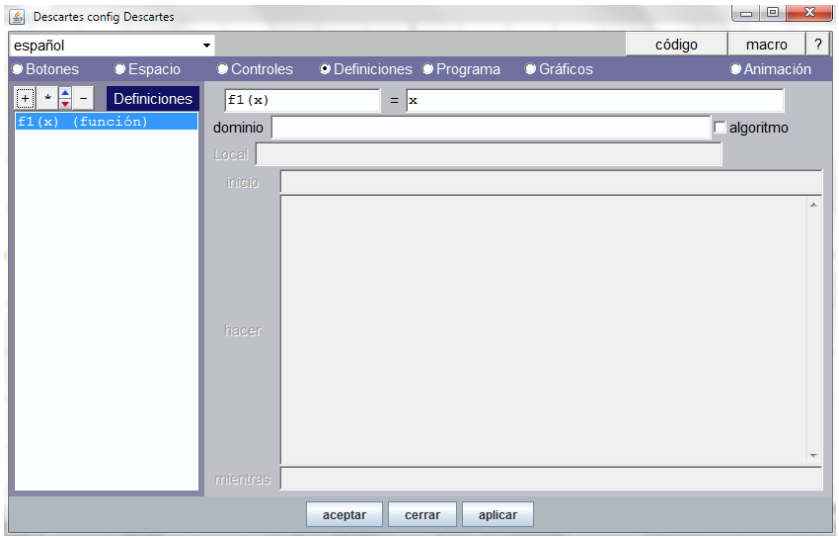

Figura 11.3: Ejemplo del tipo de definición *Función*.

- **identificador de la función**: es un campo de texto en el que se introduce el nombre por el cual se ha de llamar la función. Las funciones pueden o no llevar parámetros cuando son llamadas. Estos parámetros se incluyen entre paréntesis al final del nombre de la función, y si se trata de más de un parámetro, éstos se separan por comas. En caso que la función no involucre parámetros, el paréntesis sigue siendo necesario, aún cuando se deje vacío.
- **valor o expresión de retorno de la función**: es un campo de texto que incluye un valor o expresión que la función ha de devolver cuando es llamada. Cuando se llama una función, muchas veces se asigna ésta a alguna variable. El valor de la expresión de retorno es precisamente el que se asignará a la variable asociada.
- **dominio**: es un campo de texto donde se introduce una condición. La función sólo se evaluará en los puntos donde la condición es verdadera.
- **algoritmo**: es un checkbox que cuando está marcado permite que la función no se calcule sólo mediante una única instrucción, sino como un grupo de instrucciones. De estar marcado, habilita los campos *Local*, *inicio*, *hacer* y *mientras* de la función, cuya funcionalidad es igual a la de los algoritmos INICIO y CALCULOS.
- **Local**: es un campo donde se introducen los nombres de las variables que se han de considerar locales. Una variable local aquí definida puede cambiar su valor sólo dentro de la función. Por ejemplo, si hay una variable i dentro de la función en cuestión, y otra variable i en alguna otra parte del programa, aunque se llamen igual, si está declarada como local, aunque se le cambie su valor en la función no se afectará el valor de la otra variable que está en la otra parte del programa.
- **inicio**, **hacer** y **mientras**: son campos de texto que se habilitan si el checkbox *algoritmo* está marcado. Su funcionalidad es igual a la del algoritmo INICIO, por lo que se puede consultar en el apartado INICIO.

Hagamos primero un ejercicio para practicar las funciones de una sola instrucción. El objeto de este ejercicio es hacer un interactivo que indique la longitud de los tres lados de un triángulo cuyos vértices son controles gráficos. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Definiciones 02. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/

Definiciones_02/Definiciones_02_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio usamos una función que sólo involucra una instrucción y que, además, devuelve un valor. Para que devuelva el valor, es necesario que la expresión que devuelve el valor esté tras el signo = de la función. En este caso, la expresión devuelta es la raíz cuadrada de la suma del cuadrado de los catetos del triángulo rectángulo cuya hipotenusa está flanqueada por los puntos (ax,ay) y (bx,by). Observamos que dicho campo puede tener un valor, una variable, o inclusive (como es el caso presente) una expresión a devolver. La función devuelve un valor que se asigna a la variable en cuestión (en este caso *distg1g2* y las otras dos distancias).

En este caso sólo calculamos las distancias entre tres puntos, pero si tuviéramos veinte puntos, la misma función se puede usar para las veinte distancias. Podemos, así pues, notar que la función sirve como una instrucción que tiene el potencial de ser usada muchas veces, aunque en el código sólo se introduce una vez.

En este ejercicio se usaron variables del tipo *g1.x* y funciones del tipo *sqrt()*. Todas estas son variables y funciones intrínsecas de Descartes y se pueden estudiar en los apartados sobre variables de los controles gráficos y funciones comunes a diversos lenguajes de programación.

Ahora hagamos un ejercicio que involucre una función que no devuelve explícitamente un valor a una variable, pero que sí calcula varias cosas a la vez. Queremos que el interactivo de este ejercicio nos indique no sólo la distancia entre un par de puntos asociados a dos controles gráficos, sino que también nos diga la componente horizontal y vertical de la hipotenusa definida entre ambos. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Definiciones 03. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Definiciones_03/Definiciones_03_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio pudimos ver que en ocasiones es necesario que una función no sólo calcule y devuelva un valor en particular. En este caso queríamos tres valores distintos. En lugar de hacer tres funciones, una para cada cálculo, se pudieron agrupar en una misma. La función bien pudo haber involucrado argumentos, pero no es necesario, ya que las variables *g1.x*,*g1.y*, *g2.x* y *g2.y* pueden ser usadas directamente en la función. Es preciso notar que esta función es útil para este caso en que sólo queremos conocer tres distancias de dos controles gráficos, pero que si hubiera de aplicarse a una gran cantidad de controles gráficos, mejor convendría usar una función general que involucre argumentos.

Es válido preguntarse por qué las instrucciones dentro de la función no se colocaron directamente en el algoritmo CALCULOS. Esto hubiera servido igual. Sin embargo, con el propósito de tener un orden un poco más claro en el código, se prefiere muchas veces hacer un código modular, donde es más fácil llamar a la función, que es el módulo que contiene las instrucciones específicas para realizar algo.

Para este ejercicio se usaron variables del tipo *g2.x* y funciones como *sqrt()* y *abs()*. Las variables de dicho tipo son intrínsecas de Descartes y se pueden estudiar en el apartado sobre las variables de los controles gráficos y las funciones, también intrínsecas, se pueden estudiar en el apartado sobre las funciones comunes a diversos lenguajes de programación.

Hagamos un último ejercicio con funciones que involucre el cálculo del máximo común divisor de dos enteros distintos de cero usando el algoritmo de Euclides. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Definiciones 04. El documento del interactivo como tal se encuentra en http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Definiciones_04/Definiciones_04_Escena.html. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio pudimos ver cómo funciona una función iterativa o recursiva. Es decir, una función que tiene un algoritmo que ha de repetirse hasta que se alcanza una condición. El conocer el máximo común divisor se puede usar, por ejemplo, para reducir una fracción a su expresión más simple dividiendo el numerador y el denominador de la misma por el máximo común divisor encontrado.

Para este ejercicio se utilizó una condición booleana en el campo *mientras* de la función. Los operadores booleanos se pueden consultar en el apartado sobre condicionales y operadores booleanos. Igualmente, las funciones tales como *ent()* se pueden consultar en el apartado sobre las funciones comunes a diversos lenguajes de programación.

11.3. Vector

En ocasiones se desea utilizar una gran cantidad de variables que guardarán un tipo similar de información. En estos casos, puede resultar engorroso crear muchas variables distintas, y conviene en su lugar definir una especie de "mueble con muchos cajones". Aunque el nombre del mueble es el mismo, cada cajón tiene un número o índice que lo identifica.

Cuando se agrega un vector, sólo es necesario dar su nombre. Por ejemplo, el vector *V1*. Una vez agregado, siempre se hará referencia al mismo indicando el número de cajón en cuestión. Por ejemplo, *V1[0]* representa el primer cajón, *V1[1]* representa el segundo cajón, y así sucesivamente. Como se puede notar, el índice de los cajones empieza siempre en cero. Cada cajón puede guardar un valor, como si fuera una variable común y corriente.

En la Figura 11.4 se observan los componentes de una definición tipo *vector*.

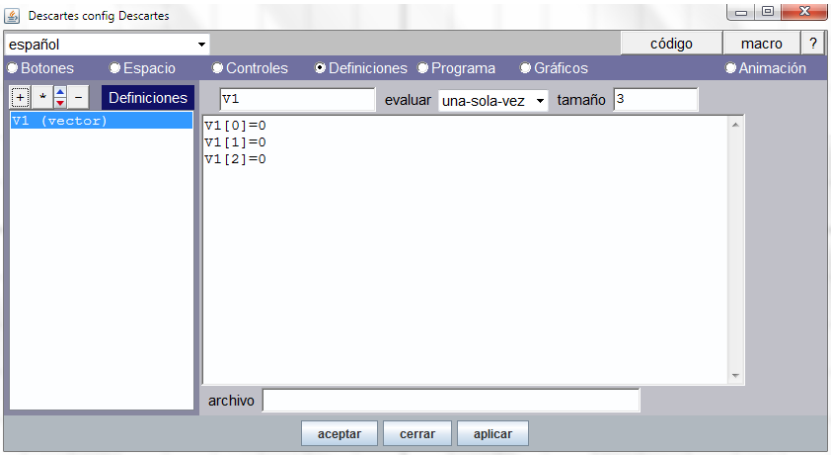

Figura 11.4: Ejemplo del tipo de definición *Vector*.

- **identificador**: es un campo de texto en el que se introduce el identificador del vector. Éste será el nombre con el que se refiere a dicho vector, y cuando se quiera introducir alguna entrada (que sería el cajón) en el programa, ésta se pone flanqueada por corchetes cuadrados después del identificador. Por ejemplo, *Vc[2]* corresponde a la tercera entrada del vector *Vc*.
- **evaluar**: es un menú con dos posibles opciones. Son las mismas opciones presentadas para los algoritmos INICIO y CALCULOS: *una-sola-vez* y *siempre*. Si se selecciona siempre, las asignaciones en el panel de texto central del vector se hacen siempre que el usuario modifique un control. De lo contrario, sólo se hacen al inicio. Dado que las actualizaciones a los valores de los vectores suelen hacerse mediante funciones, conviene dejar esta opción en *una-sola-vez* la mayoría de las veces.
- **tamaño**: es un campo de texto en el que se introduce el valor del número de entradas (o cajones) del vector.
- **panel de asignaciones del vector**: es un panel para introducir el texto de las asignaciones del vector. Por defecto aparecen las 3 primeras entradas del vector inicializadas en cero. Pero

también puede incluir asignaciones a otras variables que no sean el vector. Si el vector ha de modificarse constantemente mediante una función y no requiere de valores iniciales particulares, entonces conviene dejar dicho panel vacío.

- **archivo**: es un campo de texto en el que antes era posible introducir la ruta relativa a un archivo del cual se obtendrá la información para llenar las entradas del vector. Dicha funcionalidad es aún vigente sólo para el editor, pero cuando se intenta usar con la escena html ya en el navegador, resulta imposible leer el archivo. Ello se debe a que las nuevas normas de seguridad no permiten que el programa tenga acceso a los archivos en el ordenador del usuario. De tal forma que dicho campo puede ser ignorado.

Ahora que ya conocemos las partes del vector, hagamos un ejercicio que tira un par de dados y el número de veces que sale un número (entre el 2 y el 12) se guarda en las entradas de un vector. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Definiciones 05. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Definiciones_05/Definiciones_05_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Este ejercicio puede usarse para mostrar cómo en un tiro de un par de dados es más probable tirar, por ejemplo, 7 que 2. Ello se debe a que un tiro de 2 sólo se logra si cada dado tiene un valor de 1 (una posible configuración), mientras que el tiro de 7 se logra con 1 y 6, 2 y 5, 3 y 4, 4 y 3, 5 y 2, y 6 y 1 (6 distintas configuraciones). Una posible extensión a este ejercicio que se deja al usuario es representar las frecuencias de los tiros usando una gráfica de barras. Ello se puede lograr usando una familia de polígonos.

En este ejercicio podemos notar varias cosas importantes. Por un lado, que un vector reduce el trabajo de programación pues evita la necesidad de agregar, en este caso, 11 distintas variables. Adicionalmente, si se hubieran usado 11 variables en lugar del vector, sería necesario un código muy largo para indicar a qué variable se le debe sumar la unidad cada vez que sale un determinado tiro.

Otra cosa interesante es el orden en que debe estar el vector en el panel *Definiciones*. Se debe encontrar sobre la función que lo llama. De otra forma, la función buscará un vector aún no definido y un error ocurrirá. Es por ello que siempre es conveniente colocar tanto vectores como matrices hasta arriba de la lista del panel izquierdo de *Definiciones*.

Por último, notamos la estrecha relación que se forma entre el uso de vectores (y matrices también) con las funciones. Las funciones son muy útiles para asignar valores a los vectores ya que ellas pueden manejar una variable que funciona como índice, y este índice puede usarse directamente para modificar el valor de una entrada de un vector.

11.4. Matriz

Se observó previamente lo poderoso que puede ser un vector para manejar grandes cantidades de información. No obstante, a veces es necesario manejar información no sólo en una hilera de datos, sino como celdas en una tabla. Es entonces cuando ocupamos las matrices.

Las matrices (a veces también llamadas *arreglos*) consisten en una especie de tabla con diversas celdas. Puede ejemplificarse con una cajonera que tiene hileras de cajones y columnas, y cada cajón guarda un determinado valor. En este caso, a diferencia de los vectores, se debe especificar la coordenada de la entrada de la matriz con un par de índices separados por una coma. Por ejemplo para una matriz *M*, *M[3,7]* correspondería a la entrada de la cuarta columna (recuerde que los índices se empiezan a contar desde el valor cero) y octava fila. Es decir, la notación es de la forma *M*[*columna,fila*] En la Figura 11.5 se observan los componentes de una definición tipo *matriz*.

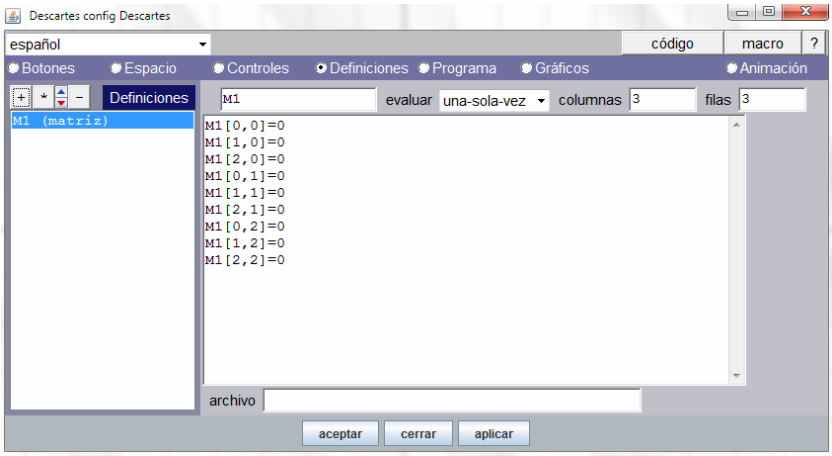

Figura 11.5: Ejemplo del tipo de definición *Vector*.

- **identificador de la matriz**: es un campo de texto en el que se introduce el identificador de la matriz. Cuando se hace referencia a alguna entrada de la matriz, se usa dicho identificador seguido del par de coordenadas de la entrada de la misma flanqueado por corchetes cuadrados. Por ejemplo, para una matriz *M*, una entrada de la misma sería *M[1,2]*.
- **evaluar**: es un menú con el mismo par de opciones que para los algoritmos INICIO y CALCULOS, y de hecho también se encuentra para los vectores. La opción *siempre* se usa cuando se desea que las instrucciones del panel de la matriz (que se encuentra abajo de este menú) se realicen cada vez que el usuario interactúa con algún control del interactivo. La opción *una-sola-vez* se usa para sólo hacer dichas instrucciones al cargarse el interactivo. Nuevamente, ésta opción es la más utilizada ya que la modificación del contenido de la matriz suele hacerse mediante funciones.
- **columnas**: es un campo de texto donde se introduce el número de columnas que tendrá la tabla. Recuerde que la tabla no tiene un *tamaño* como el vector, pues no es de una sola dimensión, sino que es de dos dimensiones. Es por ello que se requiere definir tanto las *columnas* como las *filas*.
- **filas**: es un campo de texto donde se introduce el número de filas que tendrá la tabla.
- **panel de asignaciones de la matriz**: es un panel de texto en el que se introducen las asignaciones o instrucciones para inicializar la matriz. Por defecto vienen algunas de las entradas (o cajones) de la matriz inicializadas en cero. No obstante, recuerde que, al igual que los vectores, las matrices pueden inicializarse mediante una función.
- **archivo**: es un campo de texto en el que antes era posible introducir la ruta relativa a un archivo del cual se obtendrá la información para llenar las entradas de la matriz. Dicha funcionalidad es aún vigente sólo para el editor, pero cuando se intenta usar con la escena html ya en el navegador, resulta imposible leer el archivo. Ello se debe a que las nuevas normas de seguridad no permiten que el programa tenga acceso a los archivos en el ordenador del usuario. De tal forma que dicho campo puede ser ignorado.

Hagamos ahora un ejercicio que muestra un ejemplo de cuándo son útiles las matrices. En este ejercicio se desea mostrar un gran número de partículas que podrían ser de aire en un espacio. Algunas serán rojas y otras azules. El color y la posición de las partículas están dadas al azar. Cada partícula tiene dos coordenadas (horizontal y vertical) en el espacio. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Definiciones 06. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Definiciones_06/Definiciones_06_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

El presente ejercicio muestra cómo las matrices facilitan la manipulación de una gran cantidad de información. De haber usado sólo vectores, hubieran sido necesarios como mínimo tres vectores diferentes (uno que guardara la coordenada horizontal, otro que guardara la vertical, y un tercero que guardara el color). ¡Y en algunos otros ejemplos es posible que ser requirieran aún más! Así pues, las matrices pueden verse como un *vector de vectores*, que permite una manipulación más ágil y compacta de la información.

En este ejercicio notamos que el tipo de dato que guarda una matriz puede ser distinto. En algunas entradas (o cajones) se guardaron números (las coordenadas), mientras que en otras se guardó texto (*azul* o *rojo*).

Observe también que fue nuevamente una función la que se encargó de asignar las posiciones a las partículas, ignorando la asignación de valores inicial en la matriz misma.

Se utilizaron condicionales para la asignación del color de las partículas. Se recuerda que dicho tema se puede estudiar en el apartado sobre condicionales y operadores booleanos. También se hicieron cambios en los colores de los gráficos. El uso de la herramienta asociada se puede consultar en el apartado sobre la herramienta de control de colores.

Capítulo 12

El selector *Animación*

Las animaciones se usan para visualizar cambios del interactivo en el tiempo, en lugar de pasar directamente de un estado del interactivo a otro. Permiten al usuario *ver cómo cambia* el interactivo conforme el tiempo transcurre. Por lo mismo, es necesario especificar al programa qué tan rápido debe moverse el tiempo en el interactivo, además de qué es lo que cambiará en cada paso de tiempo.

Adicionalmente, las animaciones pueden ser cíclicas (cuando llega al final de la animación, ésta vuelve a empezar), pueden iniciar desde que se carga la escena o hasta que el usuario modifique algún control. También es posible desplegar al usuario un control para controlar manualmente la animación.

Las animaciones funcionan de la misma manera que los algoritmos INICIO y CALCULOS, y las funciones. Es decir, cuentan con un campo para asignaciones iniciales, un campo para asignaciones recurrentes o cíclicas, y un campo en donde se da la condición para detener la animación.

En la Figura 12.1 se muestran los elementos del selector en cuestión.

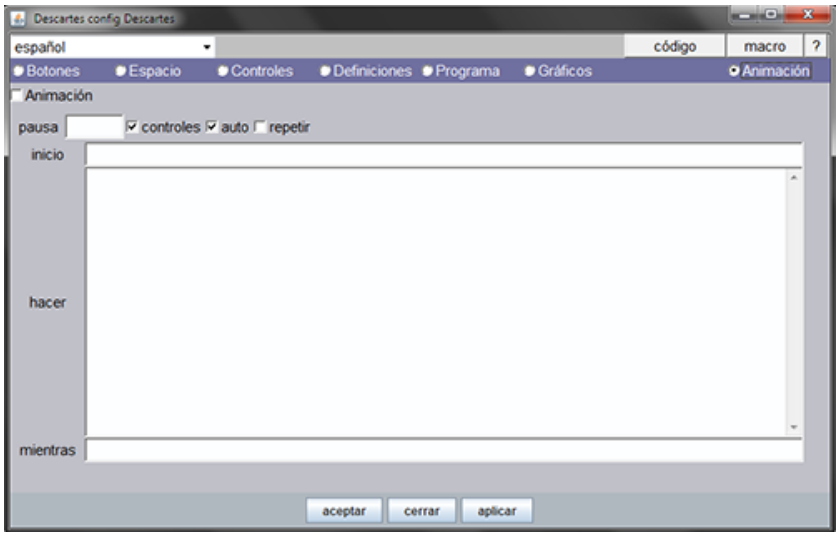

Figura 12.1: Visualización del selector *Animación*.

- **Animación**: es un checkbox que habilita todos los parámetros de de la animación descritos a continuación. Si no se encuentra marcado, es imposible editar los demás parámetros.
- **pausa**: es un campo de texto en el que se introduce un valor o una expresión que determina el número de milisegundos de la pausa entre cada paso de la animación. Es preciso notar que si la

animación involucra una gran cantidad de cálculos complejos que excedan el tiempo indicado en *pausa*, dicho tiempo no se respetará.

- **controles**: es un checkbox que cuando está marcado hace que se muestren los controles de la animación en el interactivo. Éstos aparecen como una barra pequeña de controles en la parte inferior del interactivo. Incluyen los botones *pausa* (que se convierte en reproducir cuando la animación está pausada), *reproducir* (para reproducir la animación), *detener*, *avanzar al final*, *retroceder al principio*, *avanzar un paso* y *retroceder un paso*. Estos controles sólo se muestran mientras la animación no ha concluido. Una vez concluida, desaparecen y reaparece hasta que la animación es lanzada nuevamente. Sólo funcionan en el editor (no se visualizan en navegador) y su función principal es permitir al programador depurar su código.
- **auto**: es un checkbox que si se deja marcado hace que el interactivo lance la animación desde el momento en que es cargado, sin previa instrucción del usuario. Es importante notar que esta funcionalidad en el editor sólo funciona cuando se oprime el botón *aceptar* en lugar de *aplicar*. Si sólo se pulsa *aplicar*, el interactivo se carga pero la animación no se lanza. Para ello es necesario presionar *aplicar*. En el navegador, las animaciones inician automáticamente desde el principio si este checkbox está marcado.
- **repetir**: es un checkbox que si está marcado hace que la animación se reinicie una vez que llega al final. Esta funcionalidad está pensada para hacer animaciones infinitas, por lo que el campo *inicio* de la animación es ignorado una vez que se llega a la condición de finalización de la animación y simplemente se continúan los cálculos indicados en *hacer*.
- **inicio**, **hacer** y **mientras**: son campos de texto cuya funcionalidad es igual a la del algoritmo INICIO, por lo que se puede consultar en el apartado INICIO.

Hagamos ahora un ejercicio que consiste en hacer un interactivo que el movimiento de las manecillas (segundero y minutero) de un reloj. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Animación 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Animacion_01/Animacion_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Este ejercicio abordó un ejemplo muy sencillo de animación, donde sólo se modifica el valor de un contador en cada paso de la animación. Hay animaciones que pueden ser muy complicadas e involucrar incluso funciones complicadas dentro de la animación misma. Pero convino primero abordar este ejemplo para familiarizarnos con las animaciones. Es importante notar que aquí el 1000 en *pausa* corresponde casi exactamente a 1000 milisegundos (un segundo). Ello se debe a que la instrucción que se hace en la animación es una asignación muy sencilla que no le toma tiempo al procesador. Cuando se usan instrucciones muy complicadas, el tiempo entre un paso y el siguiente de la animación puede bien no corresponder al tiempo en *pausa*, sino ser más grande puesto que además de la pausa se incluye el tiempo de procesamiento.

En este ejercicio se usaron funciones como el seno y el coseno. Pueden consultarse en el apartado sobre las funciones comunes a diversos lenguajes de programación. Igualmente, se usó una condición en la animación. Así pues, se puede consultar el apartado sobre condicionales y operadores booleanos. Finalmente, también se modificaron los colores de las manecillas. El control de colores se puede consultar en el apartado sobre la herramienta de control de colores.

Hagamos otro ejercicio que involucra animaciones un poco más complicadas. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Animación 02. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Animacion_02/Animacion_02_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Para no empezar desde cero, comenzamos usando el interactivo que resultó del ejercicio sobre matrices (Definiciones_06.html). Puede usar este vínculo para referirse a dicho ejercicio y revisarlo antes de comenzar el nuevo.

La idea de este interactivo es que las partículas ahora no sólo se muestren al inicio, sino que cada una se mueve simultáneamente como en un movimiento browniano (que vibren como si fueran partículas de polvo en el aire).

En este ejercicio se pudo notar que la animación puede incluir varias instrucciones en su interior, y algunas de estas pueden inclusive ser funciones cíclicas. Se observó que el incluir las instrucciones iniciales de la función, que mueven a las partículas una por una, pueden ser incrustadas en una función llamada por la animación para lograr que todas se muevan simultáneamente. La función, en cada paso de la animación, mueve todas y cada una de las 500 partículas.

En este ejercicio se usaron condicionales tanto en los campos *mientras* de la animación y la función, así como para el desplazamiento de cada partícula. Se puede consultar el apartado sobre condicionales y operadores booleanos para mayor información al respecto.

Capítulo 13

Funcionalidad intrínseca de Descartes

Descartes cuenta con muchas funciones y variables propias que evitan que el usuario tenga que programar las acciones más básicas. Así pues, el usuario puede bien usar las ya existentes para facilitarse el trabajo.

Es difícil tener en mente todas estas funciones y variables. Por lo mismo, se espera que este apartado funciones como una guía de rápido acceso a las funciones y variables intrínsecas de Descartes.

También se incluyen, cuando se considera importante, algunos ejercicios que agrupan el uso de diversas de estas variables, al igual que se hizo en apartados anteriores.

13.1. Variables intrínsecas de Descartes

Descartes tiene muchas variables asociadas a propiedades de los espacios, a acciones del mouse, a estatus de controles gráficos, etc. Estas variables pueden usarse tanto para conocer el estado del interactivo (por ejemplo, el tamaño de un espacio), así como para modificar propiedades mismas del interactivo. A continuación se enuncian en distintos apartados dependiendo de a qué tipo de acción o funcionalidad pertenecen.

13.1.1. Variables de Espacio

Las variables de espacio permiten conocer el alto, ancho y escala y de un espacio. Adicionalmente, el usuario puede modificarlas para lograr espacios de características particulares.

Las variables de espacio empiezan siempre con un prefijo que es el identificador del espacio en cuestión. Como notación usamos <Nombre del espacio>para indicar que ahí va el identificador del espacio.

- **<Nombre del espacio>._w**: Esta variable permite conocer el ancho de un espacio en pixeles. El sufijo *w* viene de *width*, que es anchura.
  Por ejemplo, la variable *Esp._w* nos permitiría conocer cuántos pixeles tiene de ancho el espacio *Esp*.

- **<Nombre del espacio>._h**: Esta variable permite conocer el alto de un espacio en pixeles. El sufijo *h* viene de *height*, que es altura.
  Por ejemplo, la variable *Esp._h* nos permitiría conocer cuántos pixeles de altura tiene el espacio *Esp*.

- <**Nombre del espacio**>**.Ox**: Esta variable permite conocer el desplazamiento horizontal del origen en pixeles. El sufijo *Ox* viene de *offset de x*. Por defecto, el origen del plano aparece en el centro del espacio. Pero se le puede asignar un desplazamiento (u *offset*) positivo si se desea moverlo a la derecha, o negativo si se quiere moverlo a la izquierda.
  Esta variable sirve para conocer tanto el desplazamiento horizontal del origen, pero también se le puede asignar un valor si se quiere dar un desplazamiento horizontal.

- <**Nombre del espacio**>**.Oy**: Esta variable permite conocer el desplazamiento vertical del origen en pixeles. El sufijo *Oy* viene de *offset de y*. Por defecto, el origen del plano aparece en el centro del espacio. Pero se le puede asignar un desplazamiento (u *offset*) negativo si se desea moverlo hacia arriba, o positivo si se quiere moverlo hacia abajo.
  Esta variable sirve para conocer tanto el desplazamiento horizontal del origen, pero también se le puede asignar un valor si se quiere dar un desplazamiento horizontal.

- <**Nombre del espacio**>**.escala**: Esta variable se usa para conocer la escala (el número de pixeles que hay entre una unidad y otra) del espacio. Por ejemplo, *E7.escala* es la variable asociada a la escala del espacio *E7*. También es posible asignarle un valor a dicha variable para forzar al espacio una escala deseada.

- <**Nombre del espacio**>**.rot.y**: Esta variable sólo es válida para espacios tridimensionales. Guarda la rotación del espacio en grados alrededor del eje *y*. Por ejemplo, *E2.rot.y* sería la variable que guarda la rotación en grados alrededor del eje *y* del espacio tridimensional *E2*. También es posible asignarle un valor a esta variable para forzar que el espacio al inicio tenga la perspectiva deseada.

- <**Nombre del espacio**>**.rot.z**: Esta variable sólo es válida para espacios tridimensionales. Guarda la rotación del espacio en grados alrededor del eje *z*. Por ejemplo, *E2.rot.z* sería la variable que guarda la rotación en grados alrededor del eje *z* del espacio tridimensional *E2*. También es posible asignarle un valor a esta variable para forzar que el espacio al inicio tenga la perspectiva deseada.

En el apartado sobre el selector Espacios se incluyen ejercicios en donde se echa mano de estas variables.

13.1.2. Variables del Mouse

Las variables del mouse se usan para identificar el estado del mouse. Por ejemplo, si su botón izquierdo ha sido oprimido, o si está siendo oprimido. También se pueden conocer las coordenadas relativas del mouse.

Para usar estas variables es preciso indicar el espacio relacionado (sobre el cual se quiere conocer el estado del mouse) mediante un sufijo, que es el identificador del espacio. Como notación usamos <Nombre del espacio>para indicar que ahí va el identificador del espacio.

- <**Nombre del espacio**>**.mouse_x**: Esta variable permite conocer la coordenada horizontal relativa al plano cartesiano en que el mouse se encuentra al momento de estar presionado su botón. Su valor se refresca sólo cuando el mouse está oprimido (si el mouse sólo se pasea sin estar oprimido, el valor de esta variable no se refresca).
  Por ejemplo, la variable *Esp.mouse_x* nos indica la coordenada horizontal del mouse en el espacio *Esp* cuando se hace clic con el mismo.

- <**Nombre del espacio**>**.mouse_y**: Esta variable permite conocer la coordenada vertical relativa al plano cartesiano en que el mouse se encuentra al momento de estar presionado su botón. Su valor se refresca sólo cuando el mouse está oprimido (si el mouse sólo se pasea sin estar oprimido, el valor de esta variable no se refresca).

Por ejemplo, la variable *Esp.mouse_y* nos indica la coordenada vertical del mouse en el espacio *Esp* cuando se hace clic con el mismo.

- <**Nombre del espacio**>**.mouse_clicked**: Esta variable vale la unidad cuando el mouse ha sido oprimido por lo menos una vez en el espacio en cuestión. De lo contrario, su valor es cero. Así pues, nos permite saber si el usuario ha hecho clic en un espacio o no.

- <**Nombre del espacio**>**.mouse_pressed**: Esta variable vale la unidad siempre que el botón izquierdo del mouse se encuentra oprimido. Si se suelta dicho botón, el valor de la variable regresa a cero.

Hagamos un ejercicio que permite entender mejor cómo funcionan estas variables y, adicionalmente, permiten usarlas para conocer otros estados del mouse. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Mouse 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Mouse_01/Mouse_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio observamos una forma de manipulación de las variables del mouse que nos sirven para conocer otros estados del mismo (como cuando se arrastra el mouse mientras está apretado su botón). Note que para lograr esto se aprovechó el algoritmo CALCULOS, que se repite cada vez que el usuario hace algo en el interactivo. Y dentro del algoritmo CALCULOS se incluyen una serie de instrucciones que hábilmente, usando una variable auxiliar *MPA*, permiten conocer nuevos estados del mouse. Analicemos que sucede en cada paso.

- La variable auxiliar *MPA* vale 0 al inicio. Cuando oprimimos el mouse se ejecuta el algoritmo CALCULOS. *MP* vale 1 (por lo que !*MP* vale 0), *MPA* vale 0 (por lo que *!MPA* vale 1). Así, *MDown* vale 1 & 1 que es 1; *MDragged* vale 1 & 0 que es 0; y *MReleased* vale 1 & !1 (o 1 & 0) que es cero. Sólo la variable *MDown* vale 1, que es lo esperado. Al final del algoritmo, *MPA* adopta el valor de 1 de *MP*.
- La variable auxiliar *MPA* hasta ahora vale 1. Al arrastrar el mouse mientras está oprimido, el algoritmo CALCULOS se ejecuta nuevamente. Ahora *MP* vale 1 otra vez (por lo que !*MP* vale 0) y *MPA* retiene su valor anterior de 1 (por lo que !*MPA* vale 0). Así, *MDown* vale 1 & !1 (o 1 & 0) que es 0; *MDragged* vale 1 & 1 que es 1; y *MReleased* vale 1 & !1 (o 1 & 0) que es 0. Sólo la variable *MDragged* vale 1, que es lo esperado. Al final del algoritmo, *MPA* adopta el valor del 1 de *MP*.
- La variable auxiliar *MPA* sigue valiendo 1. Cuando se suelta el botón del mouse, el algoritmo CALCULOS se ejecuta nuevamente. Ahora *MP* vale 0 (por lo que !*MP* vale 1) y *MPA* vale 1 (por lo que !*MPA* vale 0). Así, *MDown* vale 0 & !1 (o 0 & 0) que es 0; *MDragged* vale 0 & 1 que es 0; y *MReleased* vale 1 & !0 (o 1 & 1) que es 1. Sólo la variable *MReleased* termina con el valor 1, como es esperado. Al final del algoritmo, se asigna el valor de 0 de *MP* a la variable *MPA*. Es decir, volvemos al valor de *MPA* inicial.

Notamos que la variable auxiliar *MPA* es la que juega un papel crucial en determinar estos nuevos estados del mouse. Con estas instrucciones simples, podemos saber si el mouse sólo está siendo oprimido, o si está siendo oprimido y arrastrado, o si se acaba de soltar. Ello resulta útil cuando se quieren manipular objetos mediante instrucciones del mouse.

Se usaron una gran cantidad de condicionales y operadores booleanos en este ejercicio. Puede consultarlos más a fondo en el apartado sobre condicionales y operadores booleanos.

13.1.3. Variables de los controles gráficos

Las variables de los controles gráficos se usan para conocer las coordenadas relativas de un control gráfico, así como para asignarle valores a las mismas. También nos permiten saber si un determinado control gráfico está siendo usado o no.

Para usar estas variables es preciso indicar el identificador del control gráfico (sobre el cual se quiere conocer información) mediante un sufijo, que es el identificador del mismo. Como notación usamos <Identificador del control>para indicar que ahí va el identificador del control gráfico.

- **<Identificador del control>.x**: Esta variable se puede usar para imprimir el valor de la coordenada horizontal del control gráfico relativa al plano cartesiano. También es posible asignarle un valor a esta variable para colocar al control gráfico en una determinada posición horizontal. Por ejemplo, la variable *g1.x* guarda el valor de la coordenada horizontal de un control gráfico con identificador *g1*.

- **<Identificador del control>.y**: Esta variable se puede usar para imprimir el valor de la coordenada vertical del control gráfico relativa al plano cartesiano. También es posible asignarle un valor a esta variable para colocar al control gráfico en una determinada posición vertical. Por ejemplo, la variable *g1.y* guarda el valor de la coordenada vertical de un control gráfico con identificador *g1*.

- **<Identificador del control>.activo**: Esta variable nos informa si un control está siendo usado o si no. En caso afirmativo, su valor es 1 y de lo contrario es 0. Por ejemplo, la variable *g1.activo* nos indica si el control gráfico *g1* está siendo usado o no.

Hagamos un breve ejercicio con un control gráfico para ver cómo funcionan estas variables. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en ControlGráfico 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/CGraf_01/CGraf_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio podemos notar el uso de las variables de los controles gráficos que nos permiten conocer las coordenadas de un control, así como su estado de actividad. Ello resulta útil muchas veces para conocer las coordenadas donde se desea colocar algo (por ejemplo, un texto). En algunas ocasiones también se desea que se haga algún cálculo cuando un control gráfico ha sido activado, lo cual se puede lograr con un evento cuya condición es el estado de actividad del control gráfico. Finalmente, puede ser necesario regresar los controles gráficos a una posición predeterminada una vez que el usuario los ha modificado, lo cual se logra asignando a sus coordenadas los valores de dicha posición.

Para este ejercicio se usaron controles gráficos. Se puede consultar más información sobre ellos en el apartado sobre controles gráficos.

13.1.4. Variables de controles de audio y video

Existe una variable **<identificador del control>.currentTime** muy parecida a la función *<identificador del control>.currentTime()* (a la que, dándole un tiempo de argumento, posiciona la reproducción en el tiempo indicado del audio o video asociado a un control). La variable (que, a diferencia de la función del mismo nombre, no lleva los paréntesis encontrados en toda función), guarda el tiempo en segundos en que se encuentra el audio o video en ese momento. Si se ha de mostrar como parte de un texto, es necesario tener activada una animación mientras se reproduce el audio o video puesto que la animación hace que constantemente se refresque el texto que muestra la variable.

13.1.5. Variables generales de Descartes

Existe solamente una variable general de Descartes, que es *rnd*. Esta variable, cuyo nombre viene de *random* (o *aleatorio*) genera un valor aleatorio real entre 0 y 1 cada vez que es llamada. A veces se quiere tener un valor aleatorio entero y no real, y definido en un determinado intervalo. Para ello es necesario asociar la variable *rnd* con algunas funciones que nos permitan tener ese comportamiento.

Hagamos un breve ejercicio cuyo propósito es generar parábolas verticales de la forma $y = ax^2 + bx + c$, donde el coeficiente a puede adoptar valores enteros entre -3 y 3 pero sin incluir el cero (si valiera cero, no se tendría una parábola sino una recta); el b puede adoptar valores enteros entre -4 y 4 incluyendo el 0; y c puede adoptar valores reales entre 0 y 3. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en VariablesGenerales 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/VarsGrales_01/VarsGrales_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Este ejercicio es muy ilustrativo en muchos aspectos. Por un lado, cada variable tiene una asignación distinta de valores.

- La variable a primero extrae un signo de una expresión y luego lo multiplica por otra expresión que es el valor entero aleatorio entre 1 y 3. Así, permite los valores -3, -2, -1, 1, 2 y 3. Note que aunque hay dos distintos *rnd* en su asignación, cada uno lleva un valor aleatorio nuevo (no son el mismo valor).
- La variable b, al no tener la restricción de no poder valer cero, se asigna como el valor -4 al que se le suma un valor aleatorio entero entre 0 y 8, de tal forma que puede adoptar valores entre -4 y +4.
- La variable c, al no tener la restricción de ser entera, no requiere la función *ent()* que devuelve el valor entero de su argumento.
- Las variables a y b conllevan un .99999 en su interior. Se deja al usuario que analice cuidadosamente qué pasaría si esa cadena de decimales de 9 no estuviera. Considere la variable b: ¿sería igualmente probable obtener un valor de +4 para esta variable que cualquiera de los otros valores? Note también que la variable c no lleva esa cadena decimal. Al no estar el argumento sujeto a la función que extrae el entero, si tuviera dicha cadena decimal, c podría valer 3.59, y no cumpliría la restricción de tener un valor máximo de 3.

Otra conclusión que se puede extraer de este ejercicio es la importancia de usar textos de depuración. Puede ser que el texto que imprime los valores de las variables no deba ser mostrado al usuario. Pero el programador puede echar mano de él para saber si su código está haciendo lo deseado. Una vez que el programa ya quedó listo, dicho texto puede esconderse o eliminarse en su totalidad.

En este ejercicio se usaron funciones como *ent()* y *sgn()* que son propias de Descartes. Se pueden consultar con mayor profundidad en el apartado sobre funciones intrínsecas de Descartes. También se puede consultar más sobre el gráfico utilizado en el apartado sobre el gráfico ecuación.

13.2. Funciones intrínsecas de Descartes

Descartes tiene una variedad de funciones propias, muchas de las cuales son las típicas utilizadas en casi cualquier lenguaje de programación. No obstante, adicionalmente tiene otras propias relacionadas a audio, video, evaluación de funciones, etc. que se revisarán a continuación. Las funciones aquí presentadas se agrupan según su funcionalidad.

13.2.1. Funciones comunes

A continuación se enlistan alfabéticamente las funciones que Descartes tiene en común con otros lenguajes de programación. La mayoría de éstas son funciones matemáticas. En ocasiones, el nombre de la función puede variar, pero su acción es la misma.

- **abs()**: Es una función que recibe un argumento y devuelve el valor absoluto del mismo. Por ejemplo, *abs(-2)* devolverá *2*.
- **acos()**: Es una función que recibe un argumento y devuelve su arcocoseno en radianes. Por ejemplo, *acos(-1)* devolverá el valor *3.141...* radianes (que son *180°*).
- **asin()**: Es una función que recibe un argumento y devuelve su arcoseno en radianes. Por ejemplo, *asin(1)* devolverá el valor *1.570...* radianes (o *90°*).
- **atan()**: Es una función que recibe un argumento y devuelve su arcotangente en radianes. Por ejemplo, *atan(1)* devolverá el valor *0.785...* radianes (que son *45°*).
- **cos()**: Es una función que recibe un argumento, lo interpreta en radianes, y devuelve el coseno del mismo. Por ejemplo, *cos(pi/2)* (recordamos que $\frac{\pi}{2}$ equivale a 90°) devolverá el valor *0*.
- **cot()**: Es una función que recibe un argumento, lo interpreta en radianes, y devuelve la cotangente del mismo. Por ejemplo, *cot(0.785398163)* devolverá un valor cercano a *1* pues *0.785398163* es cercano a *45°*.
- **csc()**: Es una función que recibe un argumento, lo interpreta en radianes, y devuelve la cosecante del mismo. Por ejemplo, *csc(pi/2)* devolverá un valor de *1*.
- **ent()**: Es una función que recibe un argumento, lo trunca (le quita los decimales) y devuelve el valor truncado del mismo. Por ejemplo, *ent(3.78)* devolverá el valor *3*. Esta variable se puede extender a una que redondea si al argumento se le suma 0.5. Por ejemplo, *ent(3.78+0.5)* devolverá el valor *4*, que es lo mismo que redondear 3.78.
- **exp()**: Es una función que recibe un argumento y devuelve el valor de la exponencial neperiana del mismo. Por ejemplo, *exp(2)* devolverá el valor de e^2, que es *7.389...*.
- **_índiceDe_()**: Es una función que recibe dos argumentos de tipo cadena de texto. El primero es el texto principal. El segundo es un texto contenido en el primero. La función devuelve un entero que es el índice (contando el primer caracter del texto principal como el cero-ésimo) en el que empieza el texto contenido. Si no encuentra el texto contenido en el texto principal, devuelve un valor de *-1*. Por ejemplo, *_índiceDe_('hola','ola')* devolverá el valor *1*. Ésta es la única función que permite un acento (en la primera letra de *índiceDe*).
- **ln()**: es una función que recibe un argumento y devuelve el logaritmo neperiano del mismo. Por ejemplo, *ln(7.389056099)* devolverá un valor cercano a *2*, ya que $e^2 = 7.389056099$.
- **_letraEn_()**: Es una función que recibe dos argumentos. El primero es una cadena de texto. El segundo es un entero que corresponde al índice de la letra (empezando a contar la primera letra como la cero-ésima) que devolverá la función. Por ejemplo, *_letraEn_('hola',2)* devolverá el texto *l*.
- **_longitud_()**: Es una función que recibe un argumento de una cadena de texto y devuelve la longitud en caracteres del mismo. Por ejemplo, *_longitud_('hola')* devolverá el valor *4*.
- **raiz()**: Hace lo mismo que *sqrt()*.
- **sec()**: Es una función que recibe un argumento, lo interpreta en radianes, y devuelve la secante del mismo. Por ejemplo, *sec(pi)* devolverá un valor de *-1*.
- **sen()**: Hace lo mismo que *sin()*.
- **sgn()**: Es una función que recibe un argumento, y de ser éste positivo devuelve el valor +1. De ser el argumento negativo devuelve -1. Es decir, extrae sólo el signo del argumento. Por ejemplo, *sgn(-3)* devolverá el valor *-1*.

- **sin()**: Es una función que recibe un argumento, lo interpreta en radianes, y devuelve el seno del mismo. Por ejemplo, *sin(pi/2)* (recordamos que $\frac{\pi}{2}$ equivale a 90°) devolverá el valor *1*.
- **sqr()**: Es una función que recibe un argumento y devuelve su valor al cuadrado. Viene de la palabra *square*. Por ejemplo, *sqr(3)* devolverá el valor *9*.
- **sqrt()**: Es una función que recibe un argumento y devuelve la raíz cuadrada del mismo. Viene de *square root*. Por ejemplo, *sqrt(9)* devolverá el valor *3*.
- **_subcadena_()**: Es una función que recibe 3 argumentos. El primero es una cadena de texto, el segundo y tercero son enteros que corresponden a índices de los caracteres de la cadena, contando el primer caracter como el cero-ésimo. La función devuelve una cadena de texto que va del índice del segundo argumento (inclusivo) al índice del tercer argumento (exclusivo). Por ejemplo, *_subcadena_('hola',1,3)* devolverá la cadena *ol*.

Existen también funciones trigonométricas hiperbólicas (seno, coseno y tangente). Basta agregar una *h* al final de cada una (*sinh()*, *cosh()* y *tanh()*).

13.2.2. Funciones del lenguaje de Descartes

Existe pocas funciones únicas de Descartes. Una de ellas es *_Eval_()*. Esta función permite generar una función a partir de texto del usuario. De esta forma, el usuario puede introducir una función y ésta se puede evaluar y graficar. La función recibe siempre un argumento que es una variable que contiene una cadena de texto. Por ejemplo, si en alguna parte se tiene una asignación del tipo *c1='sin(x)'*, *_Eval_(c1)* estará asociada a la función seno. De esta forma, Descartes no se encuentra restringido a, por ejemplo, trazar funciones sólo dadas por el programador, sino que permite al usuario ingresar sus propias funciones.

Hagamos un breve ejercicio para dejar más claros estos aspectos. Este ejercicio consistirá en, por un lado, mostrar que el usuario puede graficar una función que él desee y, por otro lado, evaluar su función para un determinado valor de la variable independiente. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en FuncionesDescartes 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/FuncsDesc_01/FuncsDesc_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

En este ejercicio hay varias cosas dignas de observación.

- Es importante notar que cuando se usa la función *_Eval_()*, su argumento debe ser texto. Ello implica que si su argumento es una variable, ésta debe ser tipo texto. Si es un control numérico de tipo *campo de texto*, éste debe ser *solo_texto* y su valor asignado debe estar siempre entre comillas sencillas para asegurar que sea sólo texto y no sea interpretado como un valor numérico. Por ejemplo, si el valor de un campo de texto es 1 en lugar de '1', pueden llegar a presentarse errores al usarlo dentro de una función *_Eval_()*.

- También es importante notar que el código es más claro y fácil de manipular cuando las evaluaciones de los campos de texto quedan asociadas a funciones.

- Nótese la diferencia entre evaluar una función completa y evaluarla en un solo punto. El gráfico traza la función *fn()* (que no tiene argumentos), mientras que el texto imprime el valor de una función evaluada *fn2(x)* que tiene como argumento a *x*. Descartes inmediatamente interpreta el valor del interior de *_Eval_()* como la *x* a usar para la función. Muchas veces se busca que el alumno introduzca una fórmula y determinar si es correcta o no. En estos casos se puede usar esta funcionalidad y comparar los valores de la fórmula del alumno, evaluados en varios valores de la variable independiente, contra los valores teóricos. Si coinciden en varios casos, se puede decir que la fórmula del alumno es correcta.

- Aunque el editor muestra el texto de error *NaN,00* (correspondiente a *Not a Number* o *No es un número*, si el interactivo se visualiza en un navegador, aparece 0,00 en lugar de *NaN,00*. En algunas ocasiones se prefiere que el usuario no reciba mensajes de error que bien puede desconocer su significado.

Para este ejercicio se usaron varias funciones a graficar. Para más detalle sobre éstas se puede consultar el apartado sobre las funciones comunes a diversos lenguajes de programación. Se puede consultar también el apartado sobre los controles numéricos tipo *campo de texto* para más información sobre ellos.

Hay otras dos funciones propias de Descartes. Una es *parent.set('var',var)*. Esta función sirve cuando un interactivo de Descartes contiene a otro mediante un espacio HTMLIFrame (véase el apartado Espacio HTMLIFrame para más información). Si se usa esta función en el espacio subordinado, se indica que el espacio principal que lo contiene recibirá en la variable *var* el valor que el subordinado tiene en su variable del mismo nombre. Sin embargo, indicar esto no hace que el espacio principal se actualice. Es ahí donde entra la otra función propia, que es *parent.update()*. Esta función suele incluirse justo después del *parent.set('var',var)* para que el espacio principal se actualice inmediatamente.

No se incluye un ejercicio al respecto debido a que todos los ejercicio incluidos en esta documentación usan dicha funcionalidad. Puede abrir cualquiera para revisarla. Nótese que, para cualquier ejercicio, el interactivo principal es el *index.html*, que contiene a dos escenas subordinadas: *<El nombre del interactivo>_Escena.html* y *<El nombre del interactivo>_Texto.html*. En el algoritmo INICIO de *<El nombre del interactivo>_Escena.html* puede notarse cómo se usa esta funcionalidad para transferir el nombre del interactivo, mediante la variable 'ttl', al interactivo principal o contenedor. Es así como el contenedor muestra el nombre de la escena en su esquina superior derecha.

Existen algunas funciones intrínsecas asociadas a controles de audio y video que comienzan con el nombre del identificador del control de audio o video. Por ejemplo, *<identificador del control>.play*() con la que se puede controlar la reproducción del archivo. Estas funciones son:

- **<identificador del control>.play()**: Cuando se llama a esta función, el archivo de audio o video asociado al identificador del control se reproduce desde el principio. Por ejemplo, si el identificador del control es *a2*, la función sería *a2.play()*.
- **<identificador del control>.stop()**: Cuando se llama a esta función, se detiene la reproducción del archivo de audio o video asociado al control. Por ejemplo, si el identificador del control es *a2*, la función sería *a2.stop()*.
- **<identificador del control>.pause()**: Cuando se llama a esta función, se pausa la reproducción del archivo de audio o video asociado al control. Si eventualmente se vuelve a reproducir, continuará donde se quedó. Por ejemplo, si el identificador del control es *a2*, la función sería *a2.pause()*.
- **<identificador del control>.currentTime(<segundo en que inicia el archivo>)**: Ésta es una función que sí maneja argumentos. Su argumento es el valor en segundos del tiempo en el que empezará a reproducir un archivo de audio o video. Por ejemplo, *a2.currentTime(5)* hará que el archivo asociado al control *a2* empiece a reproducirse desde el segundo 5, y no desde el principio.

13.3. Condicionales y operadores booleanos

Los operadores booleanos permiten hacer comparaciones entre valores de elementos de Descartes y devolver el valor 1 o 0 dependiendo del resultado de la comparación. Los elementos que se comparan pueden ser constantes, variables, e inclusive valores de retorno de funciones.

Cuando el resultado de comparación es verdadero, se arroja un valor de 1, y cuando es falso, se arroja un valor de 0. A continuación se muestran los operadores booleanos en Descartes.

13.3.1. Los operadores booleanos y su uso en condicionales

A continuación se presenta una lista de cuáles son los operadores, así como la forma en que se usan para comparar valores.

- ==: El operador == (dos signos de igualdad uno tras otro) compara el elemento antes del operador con aquél después. Si los valores de estos elementos son iguales se considera que el resultado de la comparación es verdadero (que arroja un valor de 1), y de lo contrario falso (que arroja un valor de 0). Por ejemplo, *(2==2)* arrojaría un valor de 1 al ser una condición verdadera, pero *(2==1)* arrojaría un 0 al ser falsa.
  Aunque Descartes tolera hacer este tipo de comparación con un solo signo de igualdad, se recomienda siempre usar el doble signo para evitar confusiones. Recuerde que un solo signo de igualdad corresponde a una asignación.

- **!**=: El operador != (una admiración de cierre seguida por un signo igual) compara el elemento antes del operador con aquél después. Si los valores de estos elementos son distintos se considera que el resultado de la comparación es verdadero (que arroja un valor de 1), y de lo contrario falso (que arroja un valor de 0). Por ejemplo, *(2!=2)* arrojaría un valor de 0 al ser una condición falsa, pero *(2!=1)* arrojaría un 1 al ser verdadera.

- <: El operador <compara el elemento antes del operador con aquél después. Si el valor de aquél antes del operador es menor que aquél después, se considera que el resultado de la comparación es verdadero (que arroja un valor de 1), y de lo contrario falso (que arroja un valor de 0). Por ejemplo, *(2<1)* arrojaría un valor de 0 al ser una condición falsa, pero *(1<2)* arrojaría un 1 al ser verdadera.

- >: El operador >es parecido al <. Si el valor del elemento antes del operador es mayor que aquél después, se considera que el resultado de la comparación es verdadero (que arroja un valor de 1), y de lo contrario falso (que arroja un valor de 0). Por ejemplo, *(2>1)* arrojaría un valor de 1 al ser una condición verdadera, pero *(1>2)* arrojaría un 0 al ser falsa.

- <=: El operador <= (un signo de *menor que* seguido de un igual) se comporta como el <, pero permite también la igualdad. Si el valor de aquél antes del operador es menor **o igual** que aquél después, se considera que el resultado de la comparación es verdadero (que arroja un valor de 1), y de lo contrario falso (que arroja un valor de 0). Por ejemplo, *(2<=2)* arrojaría un valor de 1 al ser una condición verdadera, pero *(2<=3)* arrojaría un 0 al ser falsa.

- >=: El operador >= (un signo de *mayor que* seguido de un igual) se comporta como el >, pero permite también la igualdad. Si el valor de aquél antes del operador es mayor **o igual** que aquél después, se considera que el resultado de la comparación es verdadero (que arroja un valor de 1), y de lo contrario falso (que arroja un valor de 0). Por ejemplo, *(3>=3)* arrojaría un valor de 1 al ser una condición verdadera, pero *(3>=4)* arrojaría un 0 al ser falsa.

- **!**: El operador ! (un signo de admiración de cierre) se aplica antes de alguna condición o valor. Lo que hace es negar el resultado de dicha comparación. Es decir, si la comparación era verdadera (dando un valor de 1), la vuelve 0 (o falsa), pero si era falsa (dando un valor de 0), la vuelve 1 (o verdadera). Por ejemplo, *!(3>4)* devolvería un 1 (verdadero), mientras que *!(3<4)* devolvería 0 (falso). Otro ejemplo más directo es *!1*, que devolvería 0, o *!0*, que devolvería 1.

- |: El operador | (o *pipe*) consiste en una barra vertical que se puede introducir normalmente oprimiendo la tecla a la izquierda del número 1 en el teclado. Separa a dos condiciones (o valores

de 0 o 1) y, si al menos una de las dos es verdadera, de estas dos condiciones se arroja un resultado de 1 (o verdadero). Por ejemplo, *(2>3)|(3>1)* arrojaría un 1 (o verdadero) pues la condición a la derecha es verdadera. Igualmente, *(2>3)|1* arrojaría 1, pero *(2>3)|(3>4)* arrojaría 0. Así, a este operador se le conoce como *or*.

- **&**: El operador & (o *ampersand*) separa a dos condiciones (o valores de 0 o 1) y, si al menos una de las dos es falsa, de estas dos condiciones se arroja un resultado de 0 (o falso). Por ejemplo, *(2>3)&(3>1)* arrojaría un 0 (o falso) pues la condición a la izquierda es falsa. Igualmente, *(2>3)&1* arrojaría 0, pero *(2<3)&(3>4)* arrojaría 1 al ser ambas condiciones verdaderas. Así, a este operador se le conoce como *and*.

Las condicionales usan condiciones para asignar valores a una variable. Digamos que se quiere asignar a la variable *a* el valor de 10 si el valor de la variable *u* es mayor o igual que 50, pero de lo contrario se le desea asignar 20.5. Entonces se debe introducir el siguiente texto:

```
a=(u>=50)?10:20.5
```

Note que la instrucción tiene primero a la variable *a* a la que se le asignará el valor. Viene seguida, como de costumbre, del signo = de asignación, pero inmediatamente después de éste hay un paréntesis dentro del cual está la condición deseada. Después del paréntesis viene un signo de interrogación de cierre. Justo después de éste viene el valor a asignar si la condición es verdadera. Después viene un signo de dos puntos (:). El valor de asignación si la condición no es verdadera viene después de este signo. Considere como otro ejemplo la siguiente asignación:

```
a=((c<2)|(d>1))?j:sqrt(k)
```

Para interpretar esta asignación, considere que las variables *c* y *d* tienen valores 1 y 0, respectivamente. *c<2* devuelve un valor de 1 (pues la condición es verdadera), pero *d>1* devuelve un 0 (pues la condición es falsa. Después se comparan los resultados de ambas condiciones. Es decir, *1|0*. Como al menos uno de los resultados de las condiciones fue 1 (o verdadero), el resultado del operador | es 1. Así, se asigna el valor después de ? a la variable *a*, que es el valor que tenga la variable *j*. Si las variables *c* y *d* tuvieran valores 3 y 0, respectivamente, entonces a *a* se le asignaría el valor de la raíz cuadrada del valor de la variable *k*.

Hagamos un ejercicio para practicar estos conceptos. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en Condicionales 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/Conds_01/Conds_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip.DocumentacionDescartes5.zip*.

En el texto mostrado en este ejercicio se puede notar el comportamiento de las condiciones unitarias (donde se compara el valor de una variable con el de otra). También se nota el comportamiento de un par de condiciones en la asignación a la variable *mostrarrojo*. Asimismo, también se vio el comportamiento de las condicionales, en las que se usa una o varias condiciones para determinar qué valor asignar a una variable dada. Por último, se puso en práctica el uso del campo *dibujar-si* cuando se quiere que un gráfico, control, etc. se muestre sólo si se cumple una condición dada.

Se pueden comparar varias condiciones a la vez usando los operadores & y |. En este caso es importante hacer un uso adecuado de los paréntesis que flanquean las condiciones para que los operadores tengan el funcionamiento deseado.

Para este ejercicio se usaron colores en varios gráficos. Puede consultar el apartado sobre la herramienta de control de colores para una descripción más profunda de este funcionamiento.

13.3.2. Uso de variables mudas para condicionar el llamado de funciones

Algunas veces se necesita usar condicionales para determinar si se llama a una función o no. No hay una forma explícita de decirle a Descartes que ejecute una función si se cumple una condición y que de

lo contrario no la ejecute. Esto ocurre frecuentemente cuando se desea ejecutar de forma condicionada funciones que no devuelven argumentos.

Para resolver este problema, se pueden usar *variables mudas*. Estas variables no sirven realmente para nada más que para permitir asociar la condicional a una función. Por ejemplo, considere un ejemplo en que se desea asignar una serie de valores iniciales a varias variables mediante una función, pero que dicha función no devuelve un valor (el campo después del signo = de la función está vacío). Una posible asignación sería la siguiente:

```
bla=(z>3)?GeneraValores():bla
```

Aquí se desea que si la variable *z* es mayor que 3, se ejecute una función *GeneraValores()*. Esta función no tiene argumentos (por eso su paréntesis se encuentran vacío). Pero además es una función tipo algoritmo que le asigna valores a otras variables. Esta instrucción le devuelve un valor a la variable *bla* que no está definido. Pero eso no importa, pues lo que nos interesa solamente es que se llame a la función, con lo que las instrucciones dentro del algoritmo de la misma se ejecutarán. Si *z* es menor o igual a 3, el valor que se le asigna a la variable *bla* es el mismo valor que tenía. En este caso, la variable *bla* es muda pues no se usa explícitamente en el programa. Sólo se usa como un medio para llamar a una función de forma condicional.

13.4. Operadores generales

Los operadores generales consisten en los signos matemáticos que se usan comúnmente para asignar valores, sumar, restar, multiplicar, dividir y agrupar, entre otros que tienen funciones más específicas. Estos operadores no sólo sirven para cuestiones matemáticas. Por ejemplo, pueden tener también la función de concatenar textos.

Los operadores pueden trabajar sobre constantes, variables, retornos de funciones y, como se mencionó, hasta textos.

- =: El operador *igual* sirve para hacer asignaciones. Las asignaciones toman el valor de una constante, o variable o expresión colocadas a la derecha del signo y se lo asignan a la variable a la izquierda del signo. Estas asignaciones pueden pasar tanto valores numéricos como cadenas de caracteres. Por ejemplo, una asignación *a=2* toma el valor de 2 a la derecha de la igualdad y se lo asigna a la variable *a*. La asignación *b=2*3* toma el valor de la expresión a la derecha del igual, que es 6, y se lo asigna a la variable *b*. Una asignación *c='hola'* toma la cadena de caracteres a la derecha del signo y se lo asigna a la variable *c*, de tal forma que si se imprime su valor, se imprimirá el texto *hola*.
  Es importante diferenciar entre el operador = y el ==. El operador == se usa para comparar valores de expresiones, y su uso se detalla en el apartado sobre condicionales y operadores booleanos.

- +: Éste es el operador *suma*, que añade los valores de variables. También sirve para concatenar cadenas de texto. Por ejemplo, *a=2+3* hace que la variable *a* adopte el valor de 5. Pero *b='te'+' saludo'* hará que la variable *b* contenga ambos textos (cada uno entre comillas sencillas) terminen concatenados, de tal suerte que si se imprime, se observará el texto *te saludo*.

- *: El operador *producto* multiplica los valores de las expresiones a sus extremos. Por ejemplo, *a=2*pi* hará que la variable *a* tenga el valor de 6.283...

- /: El operador *división* devuelve el cociente entre dos números. Por ejemplo, si se tiene una variable *a* que vale 3 y una *b* que vale 1.5, la asignación *c=a/b* hará que la variable *c* termine con el valor 2.

- ^: El operador *potencia* toma el elemento a su izquierda y lo eleva una potencia que es su elemento a la derecha. Por ejemplo, *q=2^3* hará que *q* termine con un valor de 8. Cabe decir que este operador se puede usar para extraer raíces más allá de la cuadrada. Por ejemplo, *r=3^(1/3)* corresponde a extraer la raíz cúbica del número 3 y asignar dicho valor a la variable *r*.
- %: Éste es el operador *módulo*. Devuelve el residuo de una división. Por ejemplo, *a=23%7* asignará el valor de 2 a la variable *a*, ya que 7 cabe 3 veces en 23 y sobra un residuo de 2.
- ( ): El par de paréntesis, como operadores, se usan para agrupar expresiones en asignaciones o condiciones en una condicional. Mediante paréntesis es posible indicarle a Descartes que ignore el orden de operaciones predeterminado y que haga algunas operaciones antes que otras. Para más información sobre el orden de operaciones, consulte el apartado sobre el orden y jerarquía de operaciones matemáticas.

Es importante tener en mente los errores que se pueden cometer al hacer asignaciones y operaciones en general. En ocasiones, se puede hacer una división donde el divisor eventualmente vale cero. En este caso, Descartes suele enviar errores en la ventana de comandos. En el editor puede aparecer un aviso de que el resultado es *NaN* o *infinito*. Otros casos que generan errores son cuando se intenta extraer una raíz par de un argumento negativo, o cuando se trata de operar variables con cadenas de texto mediante los operadores distintos a + (que sirve para concatenar). Otros errores se producen cuando se usan incorrectamente paréntesis de agrupación. Si por cada paréntesis de apertura no hay uno de cierre, o sobran paréntesis de cierre, Descartes informará que hay un problema.

13.5. Orden y jerarquía de operaciones matemáticas

Al igual que cualquier otro lenguaje de programación, o inclusive cualquier lenguaje de calculadora, Descartes sigue un orden predeterminado de operaciones. Cuando hay varias operaciones en una instrucción, Descartes primero atiende las potencias, luego las multiplicaciones y divisiones, y finalmente las sumas y restas, todo en orden de izquierda a derecha. Considere la siguiente expresión.

```
a=3+2*3+4/5^2
```

Primero se aplican las potencias, por lo que la expresión se reduce a

```
a=3+2*3+4/25
```

Posteriormente se aplican los productos y divisiones, de derecha a izquierda, con lo que queda primero

`a=3+6+4/25`, y luego `a=3+6+0.16`

Después se aplican las suma y restas, de derecha a izquierda, con lo que queda primero

`a=9+0.16`, y luego `a=9.16`

Así pues, muchas veces es necesario indicarle a Descartes que haga las operaciones de forma distinta. Esto se logra usando paréntesis para agrupar las operaciones. Por ejemplo, si se desea extraer la raíz quinta de 32 ($\sqrt[5]{32}$), ella se obtiene usando $32^{\frac{1}{5}}$. Un error común en este caso suele ser la expresión

`32^1/5`, en lugar de `32^(1/5)`

La primera expresión primero hace la potencia, con lo que se obtiene

`32/5`, que finalmente da `6.4`

Así, es preciso usar el paréntesis para indicar el orden de operaciones. Si se usa

```
32^(1/5)
```

Descartes primero hará lo que está en el paréntesis que da 0.2 y luego elevará 32 al exponente 0.2, con lo que se obtiene el valor de 2 deseado.

Así, las reglas de orden de operaciones cambian al incluir paréntesis. Lo primero que se resuelve en la expresión son los paréntesis, y si dentro de ellos hay operaciones, éstas se resuelven ahí dentro de la manera convencional. Una vez que se resolvió un paréntesis y se tiene un valor que representa a toda esa expresión, se aplican las siguientes operaciones en el orden tradicional.

Es importante notar que se pueden anidar paréntesis dentro de otros. Por ejemplo, considere la siguiente expresión

```
2*(1+3/(2+1))
```

Descartes primero se encuentra el paréntesis externo. dentro de él busca si hay más paréntesis y se encuentra el interno, que es el primero que resuelvo, de donde se obtiene

```
2*(1+3/3)
```

De ahí, sigue operando el interior del paréntesis externo. Busca primero si hay productos o divisiones y las resuelve. obteniéndose

```
2*(1+1)
```

De ahí pasa a hacer las sumas y restas en ese paréntesis, de donde se obtiene

```
2*2
```

Y de ahí finalmente se tiene el valor de 4.

Así, los paréntesis permiten una mayor flexibilidad al indicarle a Descartes cómo se desea hacer las operaciones. Muchas veces se cometen errores pues a un paréntesis de apertura no corresponde uno de cerradura o viceversa. Este tipo de errores se reportan en la ventana de comandos de Descartes. Una práctica útil para evitar cometer este tipo de errores es que cada vez que se abre un paréntesis, primero se cierre y después se introduzca su contenido.

Capítulo 14

Herramientas generales

Algunas herramientas no pertenecen a un selector en particular, sino que se encuentran en una gran variedad de los mismos. Estas herramientas se discutirán en este capítulo.

14.1. Herramienta del control de colores

Esta herramienta consiste en una ventana que es lanzada cuando se pulsa el botón de color para algún objeto en Descartes. Ejemplos de objetos que utilizan colores son: una curva, el color de los ejes, el color de los textos, el color del contorno y fondo de polígonos, así como muchos otros más. En la Figura 14.1 se muestra la ventana para el control de colores.

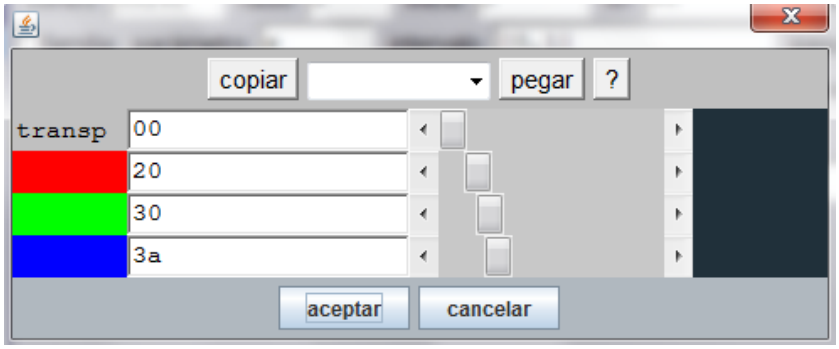

Figura 14.1: Ventana de la herramienta de control de colores

Esta ventana cuenta con los siguientes elementos:

- *copiar*: un botón que permite copiar el color actual en la ventana para pegarlo a otros elementos en el interactivo. Es útil cuando muchos objetos han de tener el mismo color y transparencia, ya que reduce el tiempo de ingresado manual de la información.
- menú de colores: un menú desplegable que permite elegir entre algunos colores prediseñados.
- *pegar*: un botón que permite pegar un color previamente copiado de tal forma que el elemento en edición termine con el mismo color que se había copiado.
- *transp*: consiste en un campo de texto y una barra horizontal. Ambos controlan la transparencia del objeto (que es qué tanto se permite ver de los objetos que se encuentran detrás del objeto siendo editado). Este control es útil cuando se prefiere dejar ver información u objetos detrás de aquél en cuestión. Es decir, cuando no se desea que el objeto cubra su fondo de forma total. Se

puede introducir en forma de texto o controlando la barra. Adelante se detalla cómo se introducen los datos en forma de texto.

- color rojo: también consiste en un control que se puede editar introduciendo texto en el campo, o bien a partir de la barra horizontal correspondiente. Este control define el componente del color rojo que tiene el objeto.
- color verde: también consiste en un control que se puede editar introduciendo texto en el campo, o bien a partir de la barra horizontal correspondiente. Este control define el componente del color verde que tiene el objeto.
- color azul: también consiste en un control que se puede editar introduciendo texto en el campo, o bien a partir de la barra horizontal correspondiente. Este control define el componente del color azul que tiene el objeto.
- panel de muestra del color: un cuadro que permite una vista previa del color del elemento editado y se encuentra a la derecha de las barras horizontales. La transparencia no se muestra en esta vista previa.

Si el color ha de introducirse en forma de texto, se puede introducir en forma hexadecimal y forma decimal, como se detalla a continuación:

14.1.1. Forma hexadecimal de introducción de color

Cuando se introduce un color en hexadecimal (también conocido como base 16), se tiene un total de 256 distintos valores posibles para cada color y la transparencia. Los valores van como sigue: 00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f, 10, 12, 13 ... 9d, 9e, 9f, a0, a1, a2 y así sucesivamente hasta el valor ff. El valor 00 corresponde al valor más bajo y el valor ff corresponde al más alto.

Cuando se introduce un valor hexadecimal en alguno de los campos de texto, se puede presionar INTRO y la barra horizontal correspondiente se actualizará. También el cuadrado de vista previa del color lo hará. De forma semejante, el arrastrar una de las barras horizontales automáticamente actualizará el valor del campo de texto y el color del cuadro de vista previa.

En caso de no estar familiarizado con la numeración hexadecimal, resulta un buen ejercicio mover una de las barras horizontales aumentando un valor a la vez para entender mejor este tipo de numeración.

14.1.2. Forma decimal de introducción de color

Aunque la notación hexadecimal es la más usada para determinar los colores, Descartes también permite una notación decimal. En este caso, el valor mínimo es el 0 y el valor máximo es el 1. Esto resulta útil cuando el color es determinado por el valor de alguna variable, en lugar de introducir sólo un valor constante determinado en forma hexadecimal. Por ejemplo, en el campo de texto de color rojo se puede poner la variable *ColorRojo*, que en algún otro lado cambia de valor según el comportamiento del interactivo. De tal forma que el cambio de color puede ser dinámico. ¿Cómo entonces convertimos entre hexadecimal y decimal?

14.1.3. Transformación de hexadecimal a decimal en la introducción de color

Muchas calculadoras científicas cuentan con un convertidor entre la numeración decimal y hexadecimal. Algunas páginas tales como http://www.binaryhexconverter.com/hex-to-decimal-converter y http://www.binaryhexconverter.com/decimal-to-hex-converter permiten estas conversiones también. Así, podemos, por ejemplo, introducir el valor hexadecimal 9a y convertirlo a decimal: 154. Dado que Descartes maneja valores en decimal entre 0 y 1, el valor obtenido ha de dividirse

entre 256, lo cual nos da un valor de 0.6015625. Introducir este valor en el campo de texto es lo mismo que introducir 9a.

La desventaja de introducir valores decimales es que, aún cuando se presione INTRO, la barra no se actualizará. Tampoco lo hará el cuadrado de vista previa de color. Pero el color será mostrado al presionar el botón *Aplicar*. En caso que se introduzca un valor menor a 0 o mayor a 1 en alguno de los colores o la transparencia, el color mostrado será negro dado que no se podrá interpretar dicho valor.

14.2. Herramienta de introducción de textos

La herramienta de introducción de textos se encuentra disponible para gráficos tales como *punto*, *texto*, así como para textos relacionados a otros elementos del interactivo más allá de los gráficos. No obstante, su funcionamiento es el mismo en todos los casos.

Normalmente se presenta un botón *texto* seguido de un campo de texto. El texto se puede introducir directamente en el campo de texto, o bien se puede oprimir el botón y elegir *Texto con formato* o *Texto simple*.

14.2.1. Introducción de texto simple

Si se presiona el botón *Texto simple* se abre una ventana donde se pueden introducir varias líneas más cómodamente. En la figura 14.2 se muestra dicha ventana.

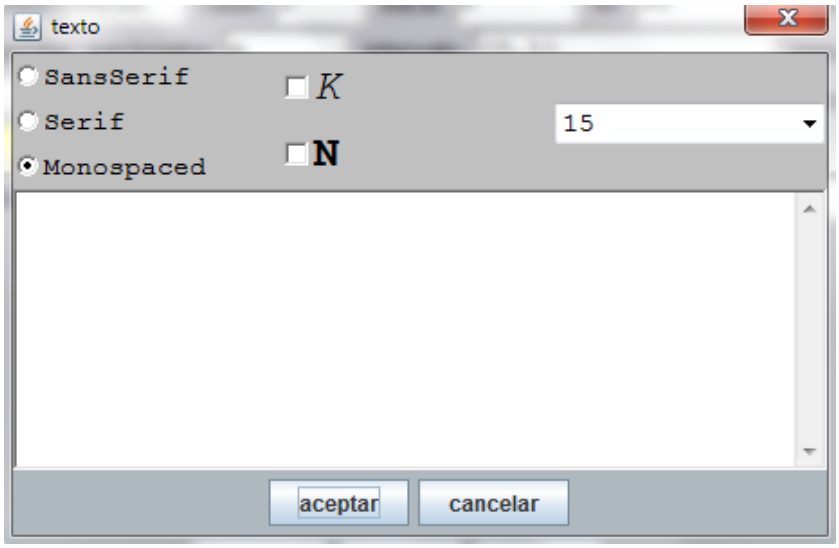

Figura 14.2: Ventana de introducción de texto simple

Es posible elegir de tres fuentes distintas para el texto: *SansSerif*, *Serif* y *Monospaced* (monoespacio). Esta fuente se aplica a todo el texto introducido en la ventana. Esto es, no es posible tener distintas fuentes en un texto simple. Cuenta además con un checkbox *K* para que el texto se muestre en itálicas y otro ***N*** para que se muestre en negritas. Por último, tiene un menú del cual se puede elegir el tamaño de la fuente. Nuevamente, estos controles se aplican por igual a todo el texto introducido en esta ventana.

Es posible introducir saltos de línea en esta ventana simplemente oprimiento *INTRO*. Una vez introducido el texto en esta ventana, se puede elegir aceptar el cambio o descartarlo con el botón *cancelar*. En caso de aceptar el cambio, el texto es trasladado al campo de texto *texto* localizado al

lado del botón que se oprimió para lanzarlo en primer lugar. El texto aparece en una sola línea en ese campo de texto. Los saltos de línea son representados por $\backslash n$ (la diagonal invertida seguida de una n). Alternativamente es también posible introducir $\backslash n$ directamente dentro de este campo de texto para que se imprima un salto de línea.

14.2.2. Introducción de texto enriquecido

Si se presiona el botón *Texto con formato*, la ventana que se muestra para la introducción de texto es diferente. Cuenta con muchos botones en la parte superior, los que permiten tener diferentes tipos de texto así como diversos símbolos matemáticos. Al hacer clic con el botón derecho del mouse en cualquiera de los controles de esta ventana se despliega una ventana con información del control al igual que en el editor de Descartes. En la Figura 14.3 se muestra dicha ventana.

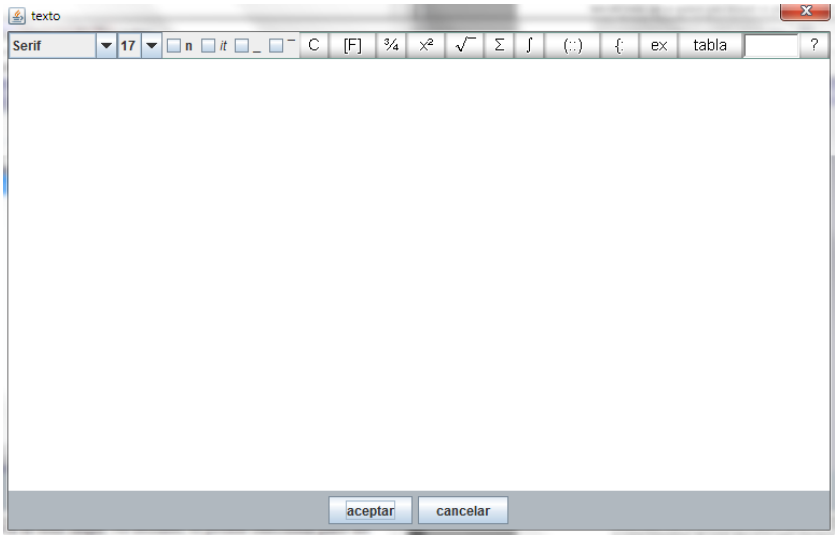

Figura 14.3: Ventana de introducción de texto enriquecido.

En esta ventana se puede introducir texto igual que en la de texto simple. No obstante, es posible seleccionar parte del texto y aplicarle cambios sólo a esa parte. Se puede usar el menú para el tipo de fuente, el menú para el tamaño, y los checkboxes de itálicas y negritas para aplicar los cambios a dicha selección.

Cuando se coloca el cursor al final del texto en esta ventana y se continúa escribiendo, algunas veces el formato de este nuevo texto no coincide con aquél a partir del cual se continuó escribiendo como ocurre en otros editores de texto. Cuando la intención es usar un texto con algunas características dadas para la ventana de texto enriquecido, es conveniente primero usar la ventana de texto simple para dar dichas características y, una vez hecho esto, editar el texto con la ventana de texto enriquecido. De esta forma suelen evitarse este tipo de problemas.

Aparte de los primeros controles en esta ventana existen algunos más sofisticados:

- : **botón subraya**: subraya el texto seleccionado
- : **botón raya superior**: coloca una raya horizontal sobre el texto seleccionado.
- **botón de color**: botón que muestra el color actual del texto. Al oprimirlo, muestra la herramienta de edición de color para asignarle un nuevo color al texto seleccionado.
- **botón [F]**: introduce a la derecha del cursor una fórmula. Esto se visualiza como una caja de bordes rosas. Una vez introducida, se puede colocar el cursor dentro de ella e introducir texto en

lenguaje matemático. Se puede saber si el cursor está dentro de una caja de fórmula pues su color se vuelve igual al del borde de la caja de la fórmula. Los 8 botones siguientes lanzan una caja de fórmula si son oprimidos cuando el cursor está fuera de una de estas cajas; y si están dentro de una, simplemente introducen el símbolo dentro del cuadro de fórmula existente. El atajo para introducir una fórmula es *CTRL + f*.

- **botón fracción**: se muestra como un botón dentro del cual hay un 3/4. Al oprimirlo introduce una fracción. Se puede colocar el cursor en el numerador o en el denominador para editarlo. El atajo para introducir una fracción es la diagonal sencilla (/) y sólo funciona cuando el cursor está ya dentro de una caja de fórmula.
- **botón elevar a una potencia**: se muestra como un botón que dentro tiene un x^2. Agrega una base y exponente. Se puede colocar el cursor en cualquiera de los dos para introducir texto en ellos. Se puede introducir este símbolo insertando el caracter ∧. Normalmente se introduce este caracter oprimiendo simultáneamente *ALT GR + {*.
- **botón extraer raíz**: tiene el símbolo de un radical en su interior. Si se oprime agrega un símbolo de radical. Se puede colocar el cursor en el radicando o en el índice del radical para su edición. Se puede introducir este símbolo con el atajo *CTRL + r*.
- **botón suma**: tiene la letra *sigma* griega mayúscula que corresponde a sumas sobre índices. Si se oprime agrega un símbolo de suma. Se puede editar colocando el cursor en su parte inferior (para indicar el índice de la suma y su valor inicial), en su parte superior (para indicar hasta donde llega el índice) y a la derecha (para indicar los términos que se habrán de sumar. El atajo para introducir este símbolo es *CTRL + s*.
- **botón integral**: al oprimirlo agrega el símbolo de una integral. El cursor puede colocarse debajo del símbolo para editar el límite inferior de la integral, arriba del símbolo para el límite superior de la misma, y a la derecha para el argumento de la integral. El atajo para insertar este símbolo es *CTRL + i*.
- **botón de matriz**: tiene en su interior un paréntesis con cuatro puntos. Abre una ventana con la cual se pueden introducir matrices de las dimensiones especificadas. El cursor se puede colocar en cualquiera de las entradas de la matriz para su edición.
- **botón definición por partes**: tiene en su interior una llave para definir funciones por partes (parecida a la llave de un cuadro sinóptico). Cuando es oprimido, abre una ventana en la que se puede introducir el número de partes en que se definirá la función. Al aceptar, aparece la llave con un número de entradas correspondiente. Se puede colocar el cursor en cualquiera de las entradas para su edición.
- **botón ex**: es un botón para introducir expresiones. En las expresiones se pueden introducir variables, constantes, o expresiones a calcular. Una vez evaluadas, lo que se imprime en el interactivo es el valor de las mismas. Por ejemplo, en alguna parte del programa se puede calcular $a = 2+1$. Si en la expresión se introduce a, el valor impreso sería 3.00. Para introducir un cálculo en la expresión, se hace doble clic sobre el *expr* que aparece en el editor de texto enriquecido. Ello desplegará una ventana en la cual se introduce el cálculo que habrá de hacerse. Las expresiones se pueden introducir en cualquier lugar del editor de texto enriquecido donde se pueda introducir texto simple en un cuadro de fórmula. Por ejemplo el límite superior de una integral puede ser una expresión que cambia según la manipulación del interactivo. Se puede introducir con un atajo siempre y cuando el cursor ya esté dentro de un cuadro de función, y el atajo es *CTRL + e*.
- **botón tabla**: abre una ventana con una variedad de símbolos UNICODE que se pueden introducir.
- **cuadro de texto UNICODE**: aparece normalmente vacío, excepto cuando el cursor se encuentra a la izquierda de algún caracter UNICODE. En ese caso, muestra el código de dicho caracter.

Cuando se agrega algún símbolo y aparece en un cuadro de fórmula, siempre es posible insertar texto matemático a la izquierda del símbolo colocando el cursor en esa posición. Adicionalmente, los símbolos (por ejemplo, la fracción, el radical, etc.) pueden eliminarse colocando el cursor a su derecha y oprimiendo la tecla de retroceso. Es decir, se comportan como caracteres tal cual. Es preciso tener cuidado con qué es lo que se borra. Por ejemplo, si se tiene una expresión matemática con una integral y se coloca el cursor a la derecha de la misma pero fuera del cuadro de fórmula, el oprimir la tecla de retroceso borrará toda la integral, y no el último caracter dentro de la caja. Si lo que se quisiera borrar es el último caracter, hay que asegurarse que el cursor esté dentro de la caja de fórmula (parpadeando del mismo color de la misma) y a la derecha del caracter que se desea borrar.

Es posible también combinar las expresiones para generar fórmulas matemáticas complicadas. Por ejemplo, se puede generar una fracción contínua del tipo $\frac{1}{1+\frac{1}{1+\ldots}}$ oprimiendo el botón de fracción cuando el cursor está en el denominador de otra fracción. Los tamaños de los textos se ajustan automáticamente según el nivel en que se encuentra el texto.

Es importante notar que si se hace cambio de formato de texto dentro de una caja de fórmula, dicho cambio se aplicará a todo el texto dentro de la misma. Es decir, no es posible tener distintos colores, fuentes, etc. dentro de una misma caja de una fórmula.

Cuando se oprime el botón *aceptar*, la ventana se cierra y se muestra de nuevo el editor con el texto editado. El campo de texto con el texto a imprimir se encuentra lleno de caracteres raros. Esto es un código para que el texto enriquecido se muestre correctamente. Ello se puede ver al oprimir el botón *aplicar*.

Hagamos un breve ejercicio practicando el editor de textos. El interactivo de este ejercicio, junto con las instrucciones para lograrlo, se encuentran en EditorTextos 01. El documento del interactivo como tal se encuentra en `http://arquimedes.matem.unam.mx/Descartes5/desarrollo/doc/Ejercicios/EdText_01/EdText_01_Escena.html`. Todos estos archivos están también disponibles en el archivo *DocumentacionDescartes5.zip*.

Se pudo observar con este ejercicio que el editor de texto enriquecido, a pesar de ser un poco más difícil de usar que el de texto simple, permite una visualización más estética de texto matemático. Adicionalmente, permite una funcionalidad más rica, valga la redundancia, como el poder determinar el número de decimales mostrados para cada expresión, en lugar de tener una sola opción para todas las expresiones del texto.

14.3. Visualización de textos en el Editor contra visualización en un navegador

Los textos mostrados en el editor de Descartes no necesariamente son idénticos a como se visualizan en el navegador. Existen algunas diferencias leves en, por ejemplo, los tamaños de las fuentes. Textos que en el editor se muestran apelmazados pueden mostrarse de forma correcta en el navegador. Igualmente, a veces el editor considera que el texto al final de una línea ya no cabe en un ancho de línea dado (por ejemplo cuando se da un valor al control *ancho* en el gráfico tipo *texto*) y lo manda a la siguiente línea, pero al visualizarse en el navegador, resulta ser que dicho texto pudo caber en una sola línea. Debido a estas diferencias, siempre se recomienda cargar el interactivo en el navegador dado que ése será el producto final que puede ver un usuario. Actualmente se está desarrollando una nueva versión de un editor de Descartes donde dicho problema se está abordando con el fin de que lo que se ve en el editor sea lo mismo que lo que se ve en un navegador.

Capítulo 15

Discursos de Descartes

Hasta ahora se ha visto la funcionalidad general de Descartes. Sabemos que los interactivos de Descartes se guardan como páginas web (html) y que pueden tener espacios, controles y gráficos. Dentro de los gráficos se puede echar mano del texto para imprimir información, instrucciones y retroalimentación para el usuario.

Sin embargo, en algunas ocasiones es preferible que el interactivo quede como un texto enriquecido que incluye interactivos en su interior. Esto permite una estructura más de tipo texto (como una página de un libro), dentro de la cual se albergan los interactivos. Es aquí donde podemos usar los Discursos, también conocidos como *Arquímedes*. Se dejó este apartado hasta el final dado que es preferible contar con toda la información general de Descartes para poder abordarlo fácilmente.

Como sabemos, el editor de Descartes se abre, por defecto, mostrando un interactivo con un espacio. Para agregar un discurso hay que usar la opción *Discurso* dentro del menú *Insertar*. Con esto, el discurso queda debajo del interactivo mostrado por defecto. Para quitar ese interactivo basta pulsar en el botón *S* a la derecha del mismo para que quede seleccionado con un borde rojo. Una vez seleccionado hay que presionar la tecla de retroceso (o backspace), con lo que el interactivo se elimina y sólo queda el discurso. En la Figura 15.1 se muestra el editor de Descartes una vez que se agregó el discurso y se eliminó el interactivo que se muestra por defecto al abrir el editor de Descartes.

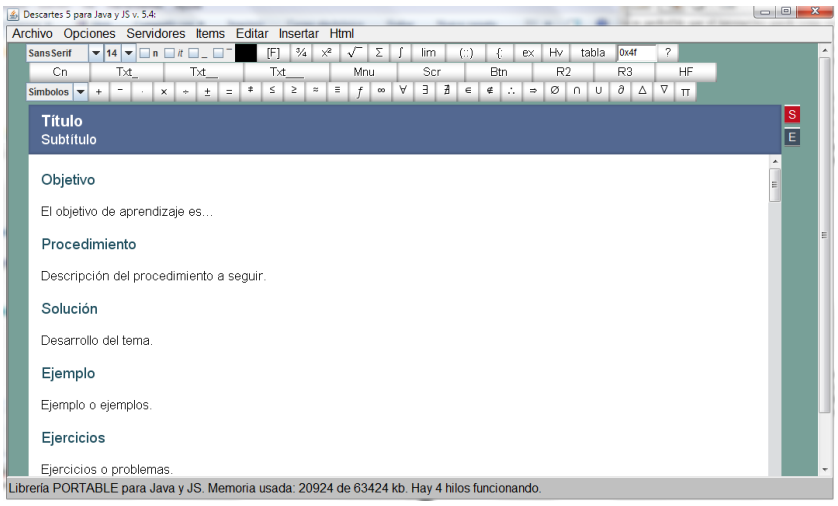

Figura 15.1: El editor de Descartes con un discurso incluido.

Cuando se trabaja con discursos hay tres hileras de botones en la parte superior del editor. La mayoría de los elementos de estas hileras ya han sido mencionados a lo largo de la documentación. Adicionalmente se presenta una barra azul para el título y subtítulo. Si se hace doble clic sobre dicha barra, se despliega una ventana emergente en donde se pueden editar estos campos.

Debajo de la barra azul aparece algunos rubros de texto, tales como el objetivo, el procedimiento, etc. Basta con hacer clic sobre alguno de ellos para editar el texto con formato enriquecido. Puede consultar esta funcionalidad en el apartado sobre la herramienta de introducción de texto enriquecido. El discurso ya trae, así pues, rubros que basta editar para crear una página.

A continuación se describen aquellos botones que no han sido abordados en la documentación general de Descartes.

- **botón Hv**: es un botón con el que se introduce un hipervínculo en donde se encuentra el cursor. Lanza una ventana en la cual se introduce una etiqueta (el texto que aparecerá como hipervínculo) y el URL (la dirección al sitio deseado). El hipervínculo en el editor no es funcional (no abre el sitio en un navegador), pero sí funciona si el interactivo se guarda y abre en un navegador.
- **botón Cn**: es un botón con el que se introduce un control numérico en donde se encuentra el cursor. Por defecto aparece como tipo pulsador.
- **botones Txt_, Txt__ y Txt___**: Son tres botones con los que se introduce automáticamente un control tipo campo de texto a la altura del cursor. El primero de los botones agrega un campo de texto corto, el segundo es mediano y el tercero es largo. Es posible modificar su tamaño después mediante el editor de configuraciones.
- **botón Mnu**: es un botón que introduce un control numérico tipo menú a la altura del cursor.
- **botón Scr**: es un botón que introduce un control numérico tipo barra a la altura del cursor.
- **botón Btn**: es un botón que añade un control numérico tipo botón a la altura del cursor.
- **botón HF**: es un botón que añade un espacio HTMLIFrame a la altura del cursor. Puede consultar el apartado sobre espacios HTMLIFrame para más información.
- **menú Símbolos**: es un menú que determina el tipo de símbolos que se muestran en el tercer renglón de botones del discurso. Permite las selecciones *Latino*, con el que se muestran las letras latinas minúsculas; emphSímbolos, con el que se muestran los símbolos matemáticos; *Griegas*, con el que se muestra el alfabeto griego en minúsculas; y *GRIEGAS*, con el que se muestra el alfabeto griego en mayúsculas.

Es importante mencionar que los controles numéricos y espacios añadidos mediante los botones en la barra superior del discurso se encuentran situados en el *escenario*. El escenario puede verse como el discurso mismo en el que se encuentra tanto texto como escenas de Descartes incrustadas. Cuando los controles están ahí, se comportan como un caracter más. Es decir, se puede insertar texto antes de ellos o después. También es posible borrarlos con las teclas de retroceso o suprimir como si fueran caracteres comunes y corrientes. Es importante revisar el editor de configuraciones después de borrar un control. En ocasiones quedan *fantasmas* de los mismos aunque ya no sean visibles en el escenario. Cuando esto ocurre, conviene eliminarlos también vía el editor de configuraciones.

En algunas ocasiones puede resultar difícil determinar cuál control en el editor de configuraciones corresponde a cuál en el escenario. Siempre se puede hacer doble clic sobre el control en el escenario y se lanzará el ya familiar editor de configuraciones teniendo seleccionado al control en cuestión. El editor de configuraciones también se puede abrir en cualquier momento con el botón *E* ubicado en el editor de Descartes a la derecha del discurso.

El texto que se inserta cuando se usa el modo discurso es como el texto con formato. No se cuenta con la funcionalidad de ajuste de línea para este tipo de texto como en los editores de texto comunes. Los saltos de línea deben, así pues, introducirse de forma manual.

Cabe recalcar también que, al igual que el texto enriquecido descrito para Descartes, es posible seleccionar alguna porción del texto en el discurso para después aplicarle algún formato (como negritas o cursivas).