Universidad Autónoma Metropolitana

Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Ingeniería en Computación

Reporte final de Proyecto Terminal:

“Algoritmo para acoplamiento automático por similitudes de imágenes digitales”

Alumno: Rojas Sandoval Miguel Ángel

Matrícula: 205205055

Trimestre: 10P

Septiembre 2010

Asesor: Dr. Risto Fermín Rangel Kuoppa

Numero económico: 27499

Tabla de contenido

Tabla de ilustraciones

1. Introducción

Este documento describe los resultados obtenidos y el desarrollo de la aplicación "Algoritmo para acoplamiento automático por similitudes de imágenes digitales", se describen los cambios o modificaciones que se hicieron a partir de la propuesta de proyecto terminal inicial así como cada una de los procesos o herramientas que se utilizaron para la elaboración de dicha aplicación.

En la primera sección se da una breve introducción sobre lo contenido en este documento, después en la segunda sección se muestra el objetivo general de la aplicación y una breve descripción de lo que se tuvo que desarrollar para lograr este objetivo.

En la tercera sección se listan los objetivos particulares del proyecto y una descripción detallada de lo que se tuvo que hacer para lograr cada uno de estos objetivos particulares.

En la cuarta sección se muestra el diagrama de clases, con las clases que se obtuvieron en la realización de la aplicación. En la siguiente sección se muestra el diagrama de estados por los que pasan cada uno de los módulos de esta aplicación mostrando tanto sus entradas como sus salidas. Y finalmente en la última se da una conclusión sobre la elaboración del proyecto.

2. Objetivo General

En base en el objetivo general que fue: "Diseñar e implementar un algoritmo para el acoplamiento de n-imágenes digitales en base a descriptores estocásticos de primer orden obtenidos a través del procesamiento digital de imágenes".

Se logró que el programa pudiera unir 5 imágenes digitales ordenadas verticalmente basándose en su textura[1] de dimensiones 640x480 pixeles, se utilizaron algoritmos de manipulación de imágenes para lograr obtener 4

[1] **Textura**, es la propiedad de las superficies externas de los objetos, que podemos percibir por medio de la vista o el tacto. La textura de un elemento es lo primero que percibimos y puede utilizarse para detectar o distinguir los diferentes objetos o regiones de una imagen.

descriptores de una región de la imagen y obtener un número n de anclas para compararlas con el 40% del lado izquierdo de la siguiente imagen por segmentos del mismo tamaño de las anclas y así buscar la mejor ancla para poder unir las dos imágenes a partir de esta ancla.

Este objetivo se obtuvo gracias a la elaboración de un software que consta de 3 módulos: el módulo de generación de descriptores, módulo buscador en siguiente imagen y el módulo combinador de imagen, los cuales se encargan de una tarea específica, cada uno de ellos requiere del cumplimiento previo del módulo anterior para su total funcionamiento en la aplicación final.

El módulo de generación de descriptores tiene como entrada la primera imagen digital, el tamaño del ancla y el número de anclas a buscar y obtiene como salida las n mejores anclas así como sus parámetros de ubicación de las anclas de la imagen.

El módulo buscador en la imagen siguiente tiene como entrada la primer imagen, las n mejores anclas de la primer imagen y la segunda imagen y obtiene como salida las anclas encontradas en la segunda imagen al ser comparadas con las anclas de la primera imagen.

El módulo combinador de imagen tiene como entrada las dos imágenes, las anclas similares de las dos imágenes que se obtuvieron en el módulo anterior y este módulo se encarga de elegir el ancla que tiene el menor porcentaje de tolerancia entre las anclas y a partir de esta anclas combina las dos imágenes obteniendo como salida una imagen combinada.

Una vez obtenida la imagen combinada el proceso se repite hasta que se analizan todas las imágenes solo que ahora la entrada del tercer módulo seria la imagen combinada en el proceso anterior más la siguiente imagen a combinar.

El objetivo general de este proyecto terminal se cumple ya que es capaz de realizar todos y cada uno de los módulos logrando así acoplar las n imágenes digitales a partir de sus mejores m anclas, obteniendo una imagen final con la combinación de las imágenes digitales si se encontró similitud entre el conjunto de imágenes ordenadas.

3. Objetivos Particulares

Involucrando cada uno de los objetivos particulares, se obtuvo un resultado satisfactorio el proyecto. Se describe cada uno de ellos, haciendo hincapié en las fortalezas y debilidades que se encontraron en la paulatina realización de estos tomando en cuenta que sin el conjunto y plena función de ellos este proyecto terminal no hubiese cumplido su fin.

A continuación se describen de manera detallada cada uno de los objetivos particulares y lo que se tuvo que desarrollar para lograr determinado objetivo.

- **Hacer un estudio comparativo para determinar los mejores 3 de 4 descriptores estocásticos de primer orden para los rangos de tolerancia de 5, 10 ,15 y 20% aplicados al algoritmo de acoplamiento automático e imágenes.**

Para desarrollar este objetivo particular lo que se hizo fue tomar las clases de los dos primeros módulos el generador de descriptores y el buscador en la siguiente imagen, las clases de estos módulos fueron modificadas para poder soportar los rangos de tolerancia de 5, 10, 15 y 20% y para poder realizar una serie de combinaciones con los descriptores estocásticos tomados en grupos de tres, por ejemplo el descriptor 1, descriptor 2 y el descriptor 3 o el descriptor 1, descriptor 2 y descriptor 4, y así sucesivamente hasta formar todas las combinaciones posibles.

Con estos cambios se desarrolló una aplicación en la que se introdujeran dos imágenes y dieran como resultado las anclas y el número de anclas encontradas en base a estos resultados se realizó un estudio y tales resultados se muestran en el archivo llamado “Estudio comparativo.pdf” que se encuentra en el mismo directorio en donde se encuentra este documento junta con la respectiva aplicación.

- **Implementar el cálculo automático de descriptores estocásticos de primer orden para regiones de tamaño arbitrario en imágenes digitales.**

Este objetivo particular corresponde al módulo de generación de descriptores de la aplicación y básicamente se subdivide en dos etapas, la primera etapa se encarga de que dada la primer imagen del conjunto de imágenes ordenadas introducidas por el usuario, calcula su equivalente de la imagen en escala de grises para poder obtener un histograma en base a sus niveles de gris ya que gracias a este se

obtienen los descriptores estocásticos, une vez obtenida la imagen en escala de grises, en la etapa dos se analiza la imagen en un 40% aproximadamente de su extremo derecho por segmentos de matrices de pixeles, cuyas dimensiones fueron introducidas por el usuario las cuales pueden ser de 9x9, 18x18, 27x27 y 36x36.

Para analizar cada una de estas matrices de pixeles se subdividen en 9 segmentos cuya dimensión depende del tamaño del ancla elegida por el usuario, para cada uno de estos segmentos se obtuvieron 4 descriptores estocásticos y se obtuvo un distancia lógica a partir del segmento del centro con los de sus extremos con la fórmula de distancia como se muestra en la Figura 1.

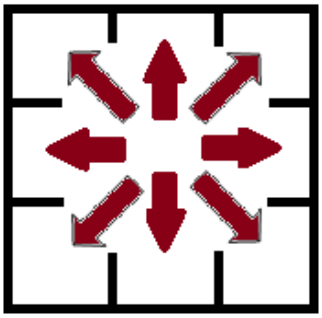


Figura 1 Segmentos de la matriz de pixeles

A continuación mostramos la fórmula de distancia lógica que se utilizó.

$$Distancia\ logica = \sqrt{(desc11 - des21)^2 + (des12 - des22)^2 \dots\dots\dots\dots.}$$

Y así sucesivamente con cada uno de los descriptores de cada segmento.

Al aplicar este procedimiento se obtuvieron 8 valores de distancias lógicas de los cuales se obtuvo un promedio y este valor se consideró como la magnitud del segmento del ancla. Al analizar el extremo derecho de la imagen se obtienen una serie de magnitudes las cuales fueron ordenadas de mayor a menor mediante el algoritmo conocido como QuickSort, las que tienen la mayor magnitud son las que son elegidas para ser consideradas como anclas porque se diferencian de los demás segmentos al tener mayor magnitud, para las cuales se seleccionan las n anclas que el usuario introdujo como dato de entrada de la aplicación.

Una de las debilidades que presenta este módulo es que entre menor sea el tamaño del ancla más tiempo se va a tardar el algoritmo en obtener las anclas porque aumenta el número de comparaciones y el número de subdivisiones por tal

motivo se introdujo el tamaño de ancla 36x36 para poder realizar las pruebas necesarias en un tiempo menor.

- **Implementar la asociación automática de regiones con descriptores similares entre una imagen fuente y una objetivo.**

Este objetivo particular corresponde al módulo buscador de la imagen siguiente del proyecto terminal. Básicamente este módulo se encarga de comparar las mejores n anclas obtenidas en el módulo anterior, con todas las posibles regiones de pixeles que se puedan obtener de la siguiente imagen, evaluándolas en el 40% del extremo izquierdo de la imagen a ser acoplada. Para lograr esta comparación se analizan las anclas obtenidas en el módulo anterior una a una con la región izquierda de la imagen a ser acoplada. Cada ancla se compara con todos los segmentos de la imagen a acoplar con las mismas dimensiones del tamaño del ancla, recorriéndola pixel a pixel.

El método de comparación que se utiliza consiste en que dadas las anclas encontradas en la primera imagen, para cada una de estas anclas se vuelve a calcular su histograma en base a sus niveles de gris y con este histograma se vuelve a obtener un vector con 4 descriptores estocásticos, la razón por la que se vuelven a calcular estos descriptores es porque en el módulo 1 se calcularon los descriptores pero de los 9 segmentos en que se dividió el ancla y para el módulo 2 se necesitaban los descriptores pero del ancla en conjunto. Una vez obtenidos los descriptores de las anclas de la primera imagen, se analiza la segunda imagen por segmentos de pixeles recorriéndola pixel por pixel, de cada uno de estos segmentos se obtienen sus descriptores con el mismos procedimiento que hasta ahora se ha descrito y se comparan con los descriptores de las anclas de la primera imagen.

De los segmentos analizados se eligen como ancla encontrada los que tengan entre 0 y 5% de diferencia entre sus descriptores y se almacenan en una lista enlazada junto con sus coordenadas en la ubicación de la imagen. Los elementos de esta lista enlazada son los que consideremos como anclas encontradas.

Uno de los problemas que se presentó en la elaboración de este Módulo es que como la segunda imagen se analiza pixel a pixel con cada una de las anclas y al tener un grado de tolerancia del 5% un ancla encontraba un número elevado de segmentos parecidos en la segunda imagen. Para solucionar este problema se tuvo que una vez que se tenía la lista se ordenaba con el algoritmo QuickSort de menor a mayor y sólo se dejó en la lista la que tuviera menor valor de porcentaje,

una por cada ancla de la primera imagen si es que se habían encontrado coincidencias, los demás elementos se eliminan.

- **Implementar la síntesis de una imagen que corresponda a la unión de dos imágenes emparejadas con los resultados de los dos pasos anteriores.**

Este objetivo particular corresponde a el *Módulo combinador de imagen* del proyecto. Esté módulo tiene como objetivo acopla las dos imágenes, en base a las coordenadas de las anclas obtenidas en el módulo anterior.

Lo que más se complicó en este módulo fue que el tamaño de la primera imagen podría variar ya que podría ser una imagen de tamaño 640x480 o una imagen ya acoplada con dimensiones diferentes a una imagen de tamaño normal. Por tal razón se tuvo que generar un algoritmo que cuando se tuviera una imagen ya acoplada se buscara la posición en la que se encuentra la imagen en donde se encontraron las mejores anclas y a partir de esta nueva posición acoplar la siguiente imagen generando una con nuevas dimensiones.

Como la imagen combinada regularmente genera una imagen final con imágenes desfasadas verticalmente en donde quedan los huecos porque no contiene algún segmento de la imagen, se rellenaron con pieles de color negro.

Una vez que el último módulo cumple su objetivo, el proceso se vuelve a repetir en dado caso de que se tengan más imágenes para ser acopladas, en caso contrario termina el proceso guardándose o no la imagen final.

4. Diagrama de clases

Este **¡Error! No se encuentra el origen de la referencia.** es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. La imagen que se presenta en este documento tal vez no sea muy explícita por falta de detalle, pero se ha incluido la imagen original *“PT Diagrama de Clases.jpg”,* donde se aprecia con mayor detalle el diagrama.

Todos los archivos antes mencionados se encuentran en el mismo directorio donde se encuentra este archivo.

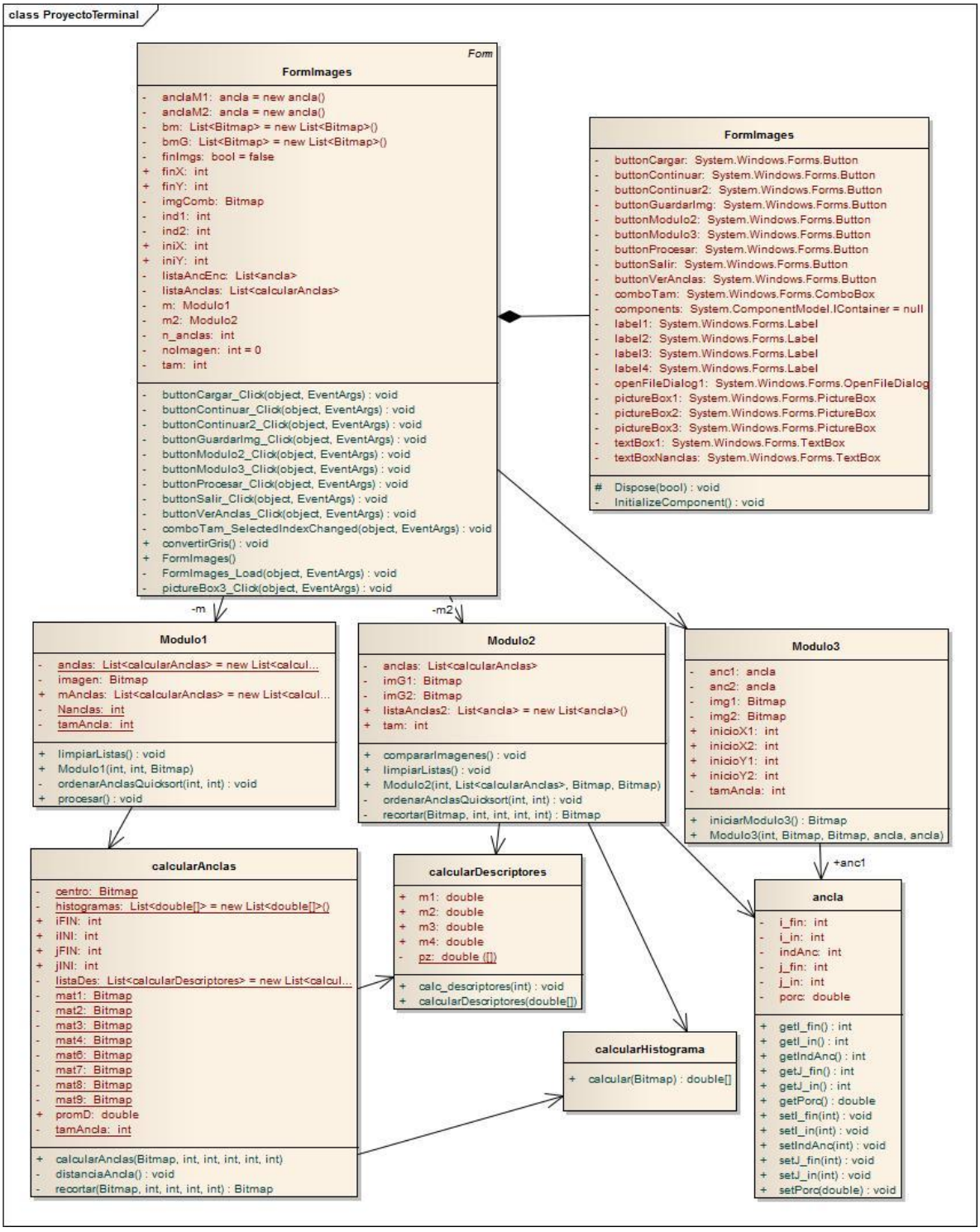


Figura 2. Diagrama de clases

5. Diagramas de estados

El **¡Error! No se encuentra el origen de la referencia.** de estados, indica cómo funcionan los módulos entre si y cuál es el orden que se sigue para lograr al acoplamiento de imágenes digitales.

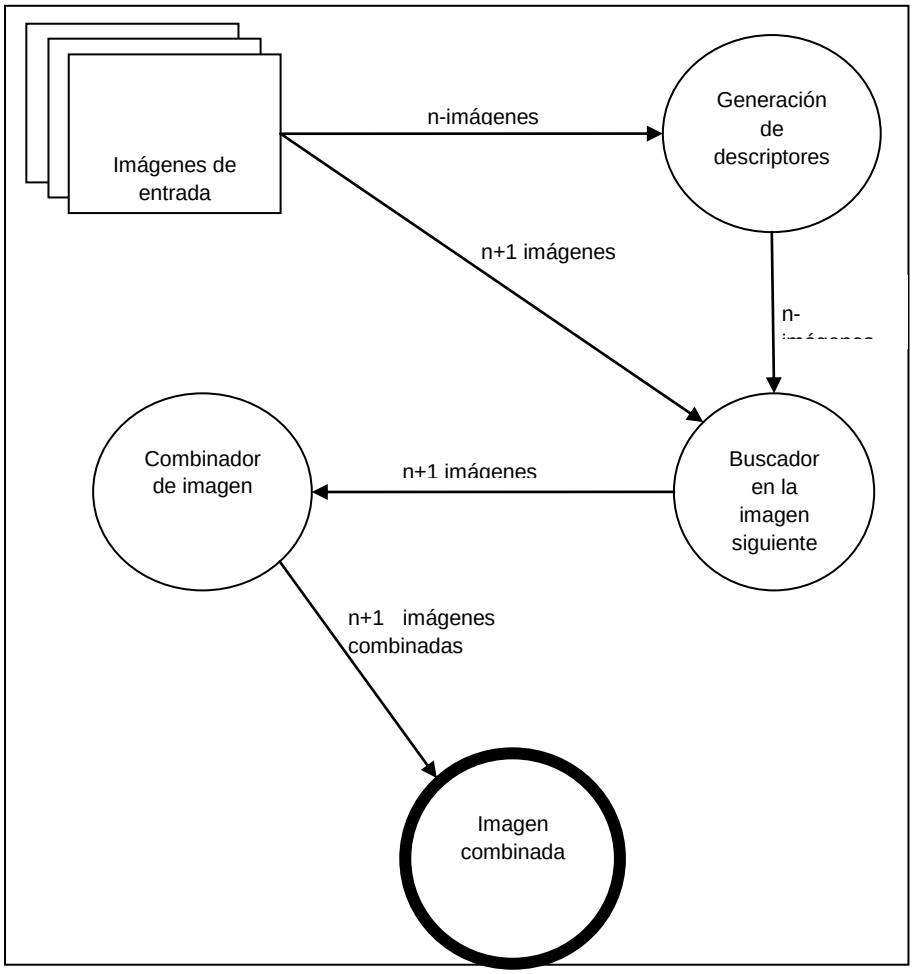


Figura 3. Diagrama de estados 1

El **¡Error! No se encuentra el origen de la referencia.**, nos proporciona información sobre las entradas y salidas de cada módulo.

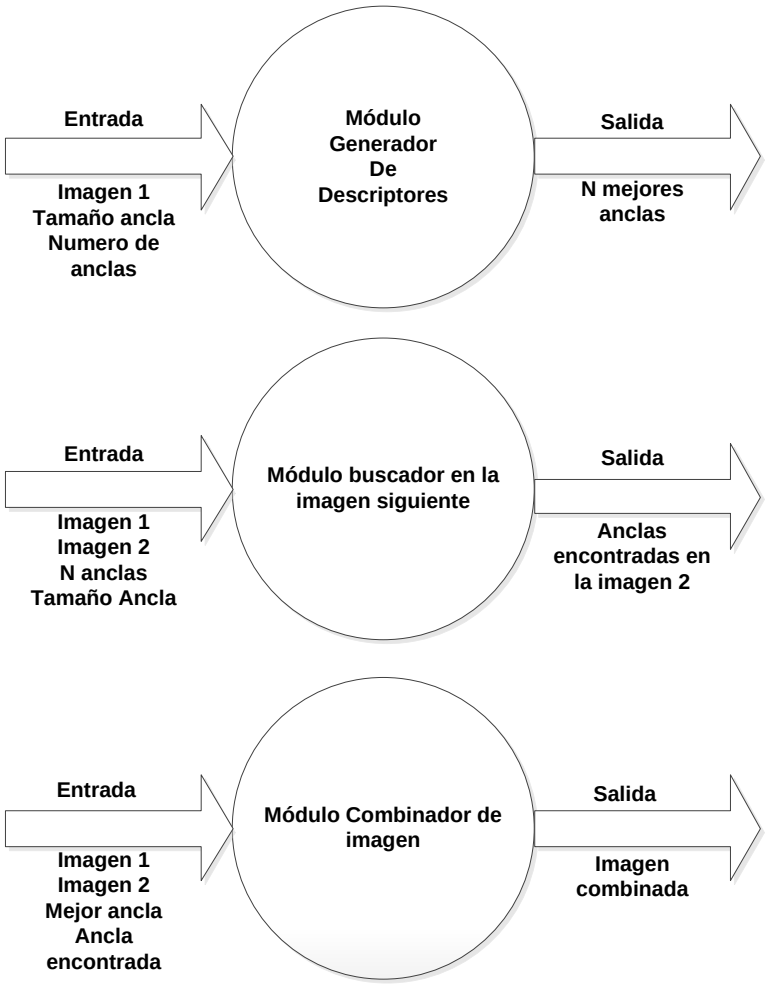


Figura 4. Diagrama de estados 2

6. Pruebas

A continuación se muestran algunas pruebas que se realizaron en la aplicación de proyecto terminal “Algoritmo para acoplamiento por similitudes de imágenes digitales”.

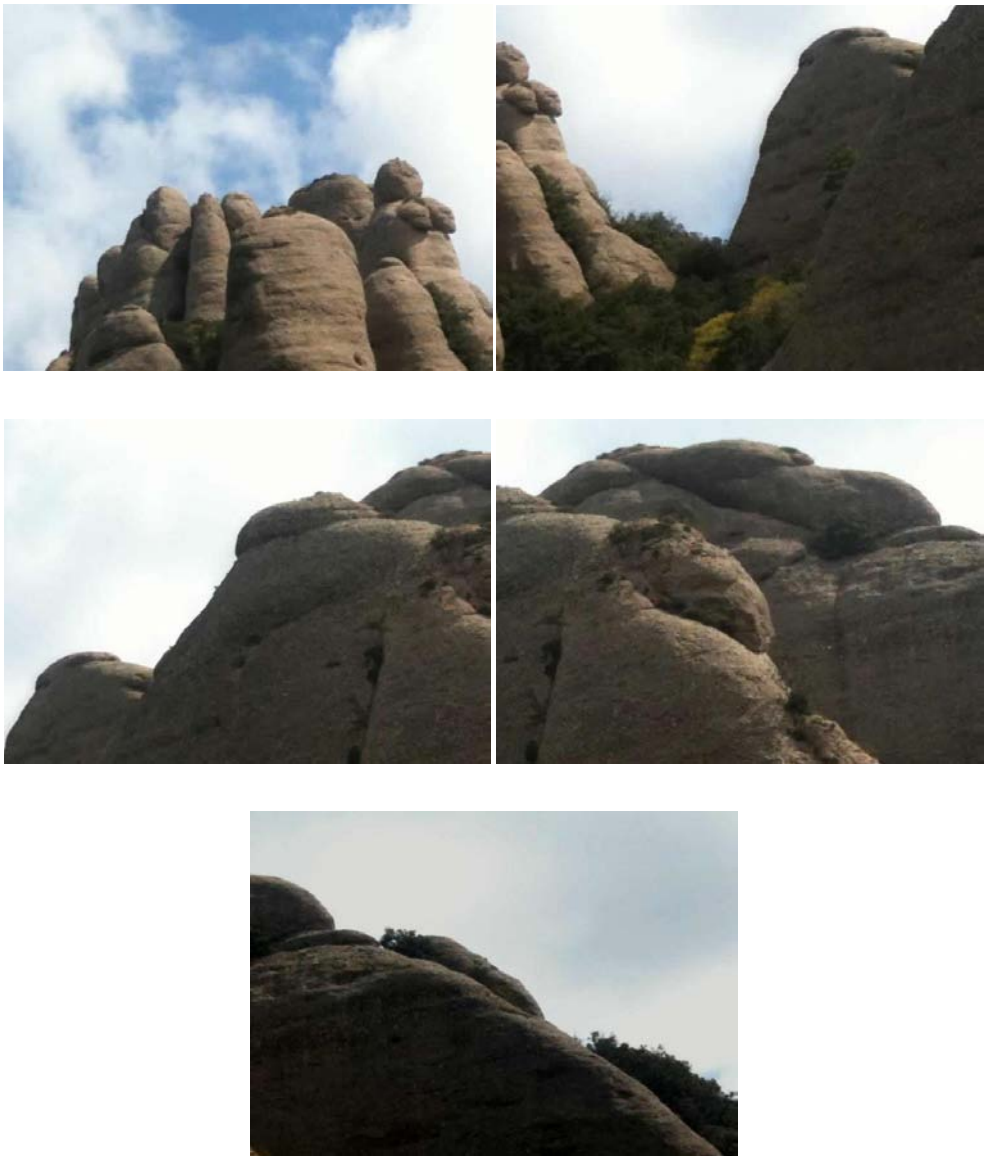


Figura 5. Imágenes de entrada

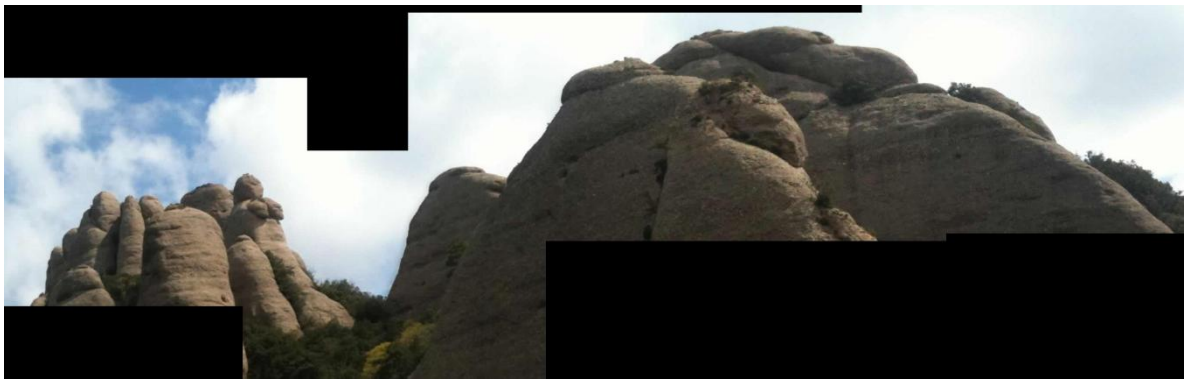


Figura 6. Imagen de salida

Figura 7. Imágenes de entrada

Figura 8. Imagen de salida

7. Conclusiones

Este proyecto presenta una implementación para acoplar imágenes digitales a partir de las similitudes entre las imágenes.

Uno de los principales problemas que se presentó al desarrollar esta aplicación fue que se intentó usar fotografías con una cámara digital común pero al momento de tomar la serie de fotografías de forma vertical al probar las fotografías digitales en la aplicación, éstas imágenes causaban mucho ruido, como por ejemplo si se tomaba alguna foto a algún objeto de cerca y al intentar mover la cámara, con el objeto visto desde otro punto cambiaba su forma, o al tomar las fotografías se tenía que hacer con una exactitud vertical, también afectaban mucho las sombras, los reflejos o el tipo e iluminación. Por tal motivo se tomaron un número elevado de fotografías en distintos paisajes hasta encontrar las adecuadas para poder probar la aplicación. Si se hubiera contado con una cámara profesional para tomar fotografías con un tripié se hubieran tomado fotografías mas exactas y no se hubieran generado tantos problemas al querer hacer pruebas en la aplicación.

Al tomar los segmentos de pixeles de las fotografías para analizarlos por ese ruido que causaban las imágenes se encontraban muchas coincidencias de anclas encontradas con una sola ancla que por lo regular no eran muy parecidas o también se daba el caso de que no se encontraban coincidencias.

Pero todos estos problemas se lograron resolver al desarrollar la aplicación obteniendo un resultado satisfactorio y cumpliendo cada uno de los objetivos.

Si en un futuro se desea retomar este proyecto para mejorarlo yo propongo mejorar los descriptores a unos de mayor orden con mayor exactitud, mejorar el algoritmo para buscar las anclas en la segunda imagen porque como se implementó en el proyecto se tenía que comparar cada ancla con el 40% del extremo izquierdo de la segunda imagen recorriéndola pixel a pixel y también se podría implementar el algoritmo de acoplamiento automático de imágenes ahora analizando la imagen total no solo en los extremos izquierdo y derecho y así obtener como resultado final imágenes más completas.

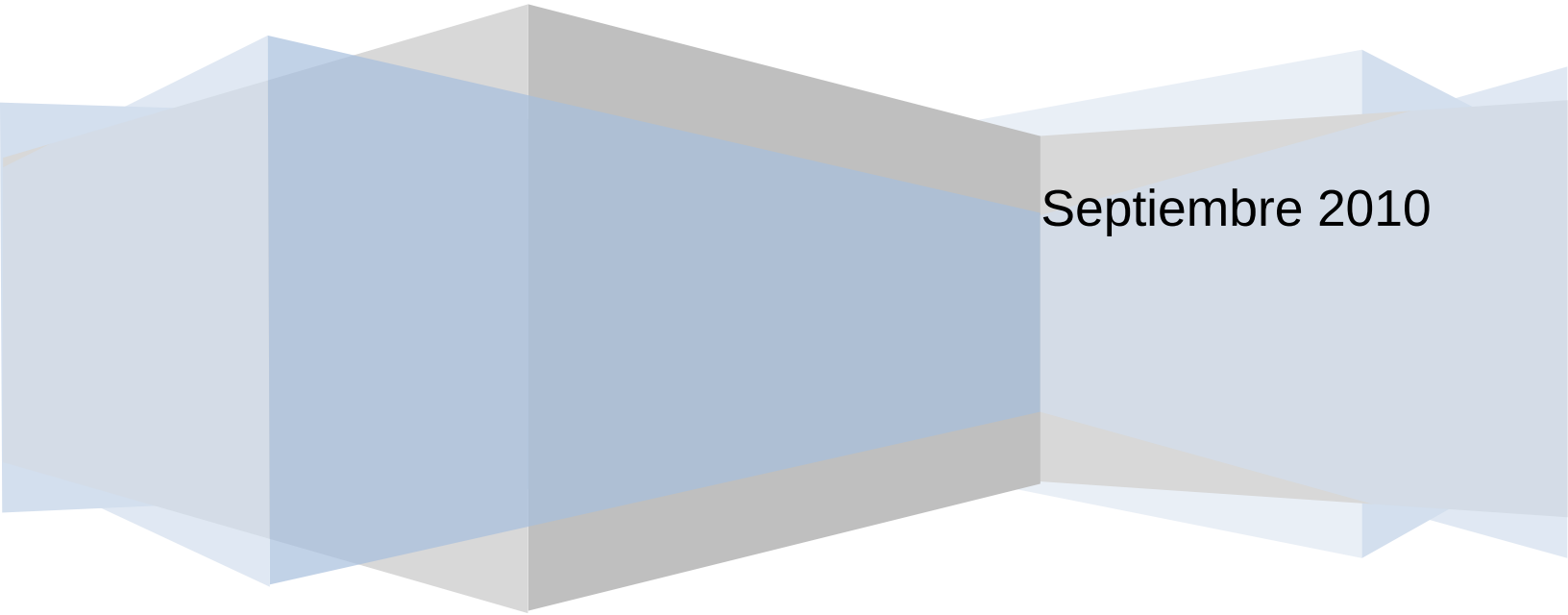
Para finalizar se puede decir que el proyecto cumple con los objetivos generales y particulares planteados en la propuesta y es capaz de acoplar una serie de imágenes tomadas con calidad y lo más exactas posible.

Universidad Autonoma Metropolitana
Unidad Azcapotzalco

“Algoritmo para acoplamiento automático por similitudes de imágenes digitales”

Estudio comparativo

Miguel Ángel Rojas Sandoval



Septiembre 2010

Estudio Comparativo

Objetivo:

Hacer un estudio comparativo para determinar los mejores 3 de 4 descriptores estocásticos de primer orden para los rangos de tolerancia de 5, 10 ,15 y 20% aplicados al algoritmo de acoplamiento automático de imágenes desarrollado para el proyecto terminal “Algoritmo para acoplamiento automático por similitudes de imágenes digitales”.

Descripción:

Para lograr desarrollar este objetivo particular lo que se hizo fue tomar las clases de los dos primeros módulos de la aplicación de Proyecto Terminal para Acoplamiento automático por similitudes de imágenes digitales, el generador de descriptores y el buscador en la siguiente imagen, las clases de estos módulos fueron modificadas para poder soportar los rangos de tolerancia de 5, 10, 15 y 20% y para poder realizar una serie de combinaciones con los descriptores estocásticos tomados en grupos de tres, por ejemplo el descriptor 1, descriptor 2 y el descriptor 3 o el descriptor 1, descriptor 2 y descriptor 4, y así sucesivamente hasta formar todas las combinaciones posibles.

Combinaciones
descriptor 1, descriptor 2, descriptor 3
descriptor 1, descriptor 2, descriptor 4
descriptor 1, descriptor 3, descriptor 4
descriptor 2, descriptor 3, descriptor 4

Tabla 1. Combinaciones de anclas

Con estos cambios se desarrolló una aplicación llamada ReporteDescriptores la cual se encuentra en el mismo directorio en donde se encuentra este archivo, esta aplicación consiste en que se introducen dos imágenes y dan como resultado el número de ancla, el número del ancla[1] encontrada, el porcentaje de diferencia entre las dos anclas y el tipo de combinación para cada ancla. Esta información se imprime en 4 archivos de texto llamados reporte5.txt, resporte10.txt, reporte15.txt y reporte 20.txt, cada archivo corresponde a los rangos de tolerancia 5, 10, 15 y 20% respectivamente para cada archivo encontradas en la carpeta Debug de dicha aplicación.

[1] **Ancla:** Podría definirse como un segmento de pixeles de una imagen digital que se diferencia de otro segmento en base a su textura

A continuación se muestra la pantalla de la aplicación:

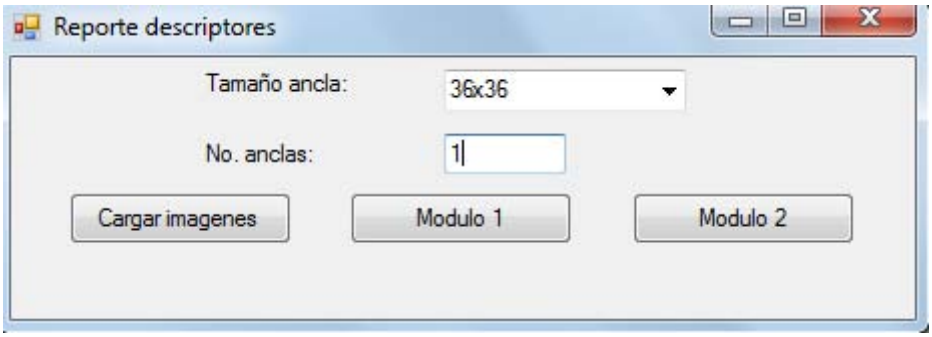


Figura 1. Pantalla de la aplicación

Procedimiento:

Para realizar las pruebas y obtener algunos resultados, se analizaron 3 pares de imágenes en la aplicación, los datos de entrada que se le introdujeron a la aplicación fueron:

No. anclas	Tamaño Ancla	Nombre de las imágenes
5	36x36	1.jpg, 2.jpg
5	36x36	DSC00328.jpg, DSC00329.jpg
5	36x36	3.jpg, 4.jpg

Tabla 2. Datos de entrada

A continuación se muestran los pares de imágenes que se analizaron.

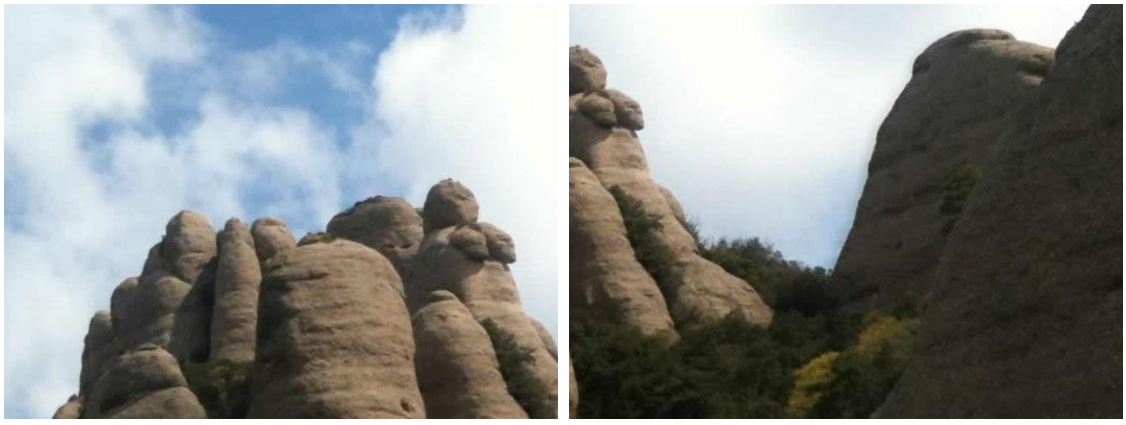


Figura 2. Primer par de imágenes

Figura 3. Segundo par de imágenes

Figura 4. Tercer par de imágenes

Resultados:

Para cada par de imágenes nos arrojó los siguientes resultados al analizarlas en la aplicación:

En donde la columna *No. Ancla* representa el número de ancla de la imagen uno, la columna *No. Encontrada* representa el número del ancla encontrada en la segunda imagen, la columna *% tolerancia* representa la diferencia en porcentaje entre las dos anclas y la columna *Descriptore*s representa la combinación de descriptores.

Para el primer par de imágenes nos mostró la siguiente información:

No. Ancla	**No. Encontrada**	**% tolerancia**	**Descriptores**
2	0	0.656675492	m1 m2 m3
3	1	3.463414775	m1 m2 m3
0	0	0.056220549	m1 m2 m4

1	1	0.009354517	m1 m2 m4
2	2	0.014630793	m1 m2 m4
3	3	0.016696174	m1 m2 m4
4	4	0.003562952	m1 m2 m4
0	0	0.066421944	m1 m3 m4
1	1	0.21562303	m1 m3 m4
2	2	0.245783485	m1 m3 m4
3	3	0.172121675	m1 m3 m4
4	4	0.041634418	m1 m3 m4
0	0	0.066458972	m2 m3 m4
1	1	0.215770281	m2 m3 m4
2	2	0.251626757	m2 m3 m4
3	3	0.173856113	m2 m3 m4
4	4	0.046292135	m2 m3 m4

Tabla 3. Resultados primer par de imágenes

Para el segundo par de imágenes nos mostró:

No. Ancla	**No. Encontrada**	**% tolerancia**	**Descriptores**
0	0	0.457950927	m1 m2 m3
1	1	0.140114591	m1 m2 m3
3	2	0.182314889	m1 m2 m3
4	3	0.273267299	m1 m2 m3
0	0	0.01864527	m1 m2 m4
1	1	0.00410971	m1 m2 m4
2	2	0.01839117	m1 m2 m4
3	3	0.004319519	m1 m2 m4
4	4	0.004231462	m1 m2 m4
0	0	0.075222355	m1 m3 m4
1	1	0.073331717	m1 m3 m4
2	2	0.259579624	m1 m3 m4
3	3	0.02990119	m1 m3 m4
4	4	0.120045354	m1 m3 m4
0	0	0.076814831	m2 m3 m4
1	1	0.076012873	m2 m3 m4
2	2	0.259547618	m2 m3 m4

3	3	0.03070662	m2 m3 m4
4	4	0.123139612	m2 m3 m4

Tabla 4. Resultados segundo par de imágenes

Para el tercer par de imágenes los resultados fueron:

No. Ancla	No. Encontrada	% tolerancia	Descriptores
2	0	0.069740275	m1 m2 m3
4	1	0.062289707	m1 m2 m3
0	0	0.008777458	m1 m2 m4
1	1	0.06800527	m1 m2 m4
2	2	0.013967767	m1 m2 m4
3	3	0.00898898	m1 m2 m4
4	4	0.002808206	m1 m2 m4
0	0	0.085918957	m1 m3 m4
1	1	0.120952302	m1 m3 m4
2	2	0.043538816	m1 m3 m4
3	3	0.053866129	m1 m3 m4
4	4	0.084311959	m1 m3 m4
0	0	0.085867702	m2 m3 m4
1	1	0.123254059	m2 m3 m4
2	2	0.04820955	m2 m3 m4
3	3	0.054768712	m2 m3 m4
4	4	0.08490268	m2 m3 m4

Tabla 5. Resultados tercer par de imágenes

Análisis de resultados:

Como puede observase en las tablas anteriores para la combinación de descriptores m1, m2 y m3, no se encontraban coincidencias para todas las anclas sólo para algunas y las que encontraba el porcentaje salía muy distinto a los de las demás combinaciones, es por eso que se descarta esta combinación.

Para las siguientes tres combinaciones en cada tabla puede verse que tienen resultados muy semejantes en las anclas encontradas, pero la combinación que sobresale de las demás es la combinación m1, m2 y m4 por que con esta combinación se encontraron anclas con un porcentaje de diferencia mucho menor que las demás, pero también podrían utilizarse las combinaciones m1, m3, m4 y m2, m3, m4 porque no generan tanto error como la primera combinación.

Para combinar las anclas en la aplicación de Proyecto Terminal para Acoplamiento por similitudes de Imágenes Digitales, se logra a partir del ancla que tenga menor porcentaje de diferencia con el ancla encontrada es por eso que las anclas de las tablas anteriores que se hubieran elegido son las siguientes.

Par de imágenes	No. Ancla	Ancla encontrada	% tolerancia	Descriptores
1	4	4	0.003562952	m1 m2 m4
2	1	1	0.00410971	m1 m2 m4
3	4	4	0.002808206	m1 m2 m4

Tabla 6. Mejores anclas

Como puede verse en la Tabla 6 todas son con la combinación m1 m2 m4 por tal razón se concluye que la combinación m1, m2 y m4 es la mejor combinación de descriptores para encontrar anclas más similares.

Conclusión:

Al realizar este estudió pude percatarme de cuál es la mejor combinación de descriptores estocásticos para el acoplamiento de imágenes a través de sus mejores anclas. Gracias a este resultado pude identificar un problema que presentaba la aplicación de Proyecto Terminal y se encontraron mejores anclas para acoplar las imágenes solo con el uso de 3 de los 4 descriptores estocásticos de primer orden.

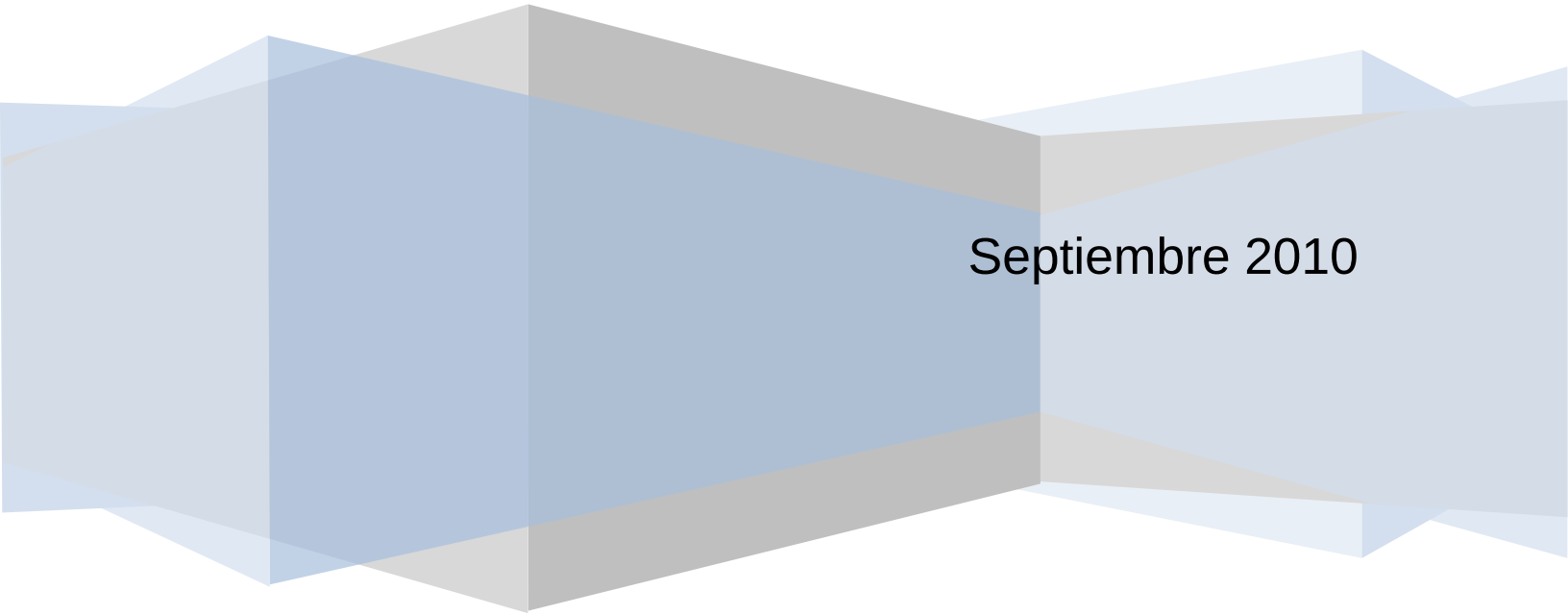
Al sólo usar estos descriptores en la aplicación se encuentra una mejor ancla para acoplar las imágenes, se ahorra tiempo al realizar menos operaciones sólo utilizando tres descriptores y sobre todo se tiene como resultado una mejor imagen final.

Universidad Autonoma Metropolitana
Unidad Azcapotzalco

Aplicación para Acoplamiento Automático por Similitudes de Imágenes Digitales

Guía rápida

Miguel Ángel Rojas Sandoval



Septiembre 2010

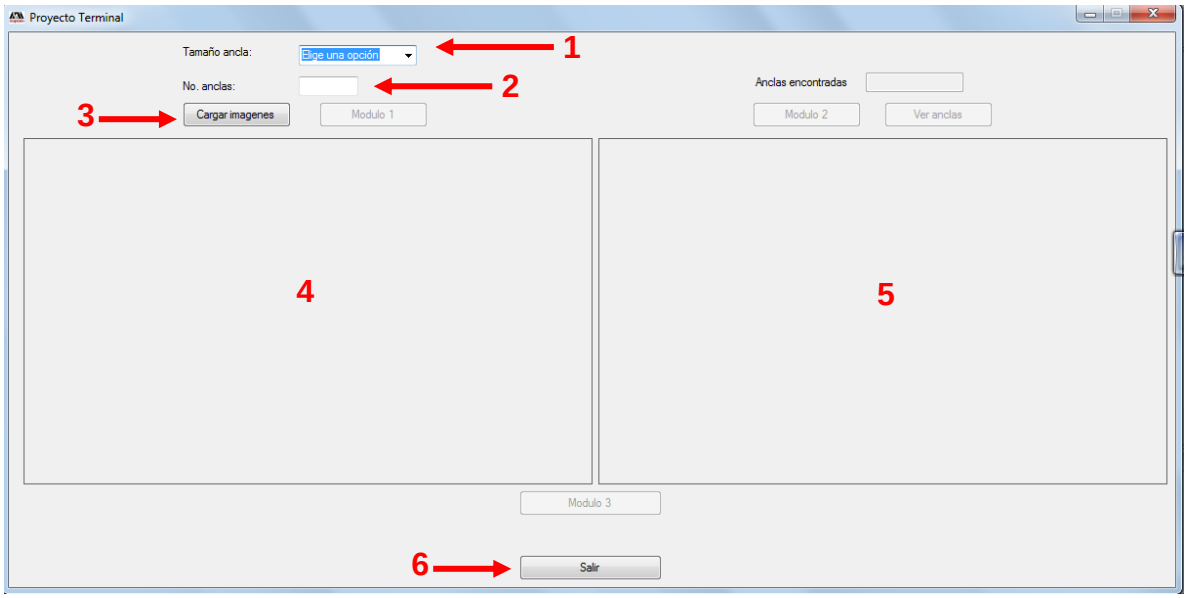
Objetivo:

El objetivo principal de ésta Guía rápida es describir de una manera simple cada uno de los componentes que integran la aplicación de Proyecto Terminal de Acoplamiento Automático por Similitudes de Imágenes Digitales.

Acoplamiento Automático por Similitudes de Imágenes Digitales

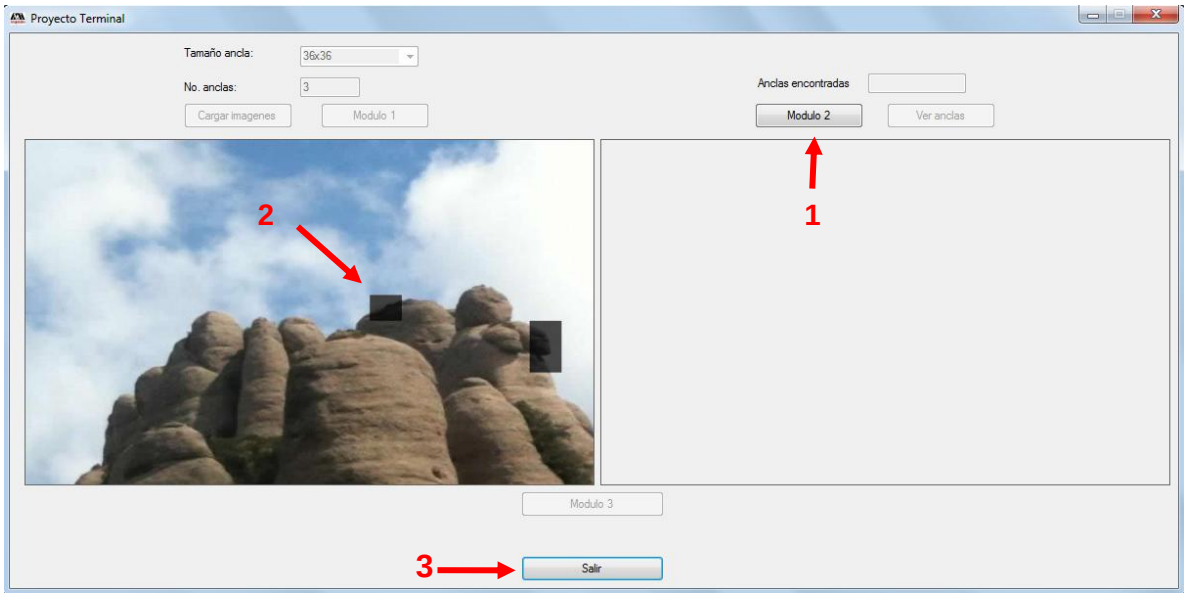
Pantalla 1: Datos de entrada y Módulo 1.

Una vez abierta la aplicación mostrará la siguiente pantalla:



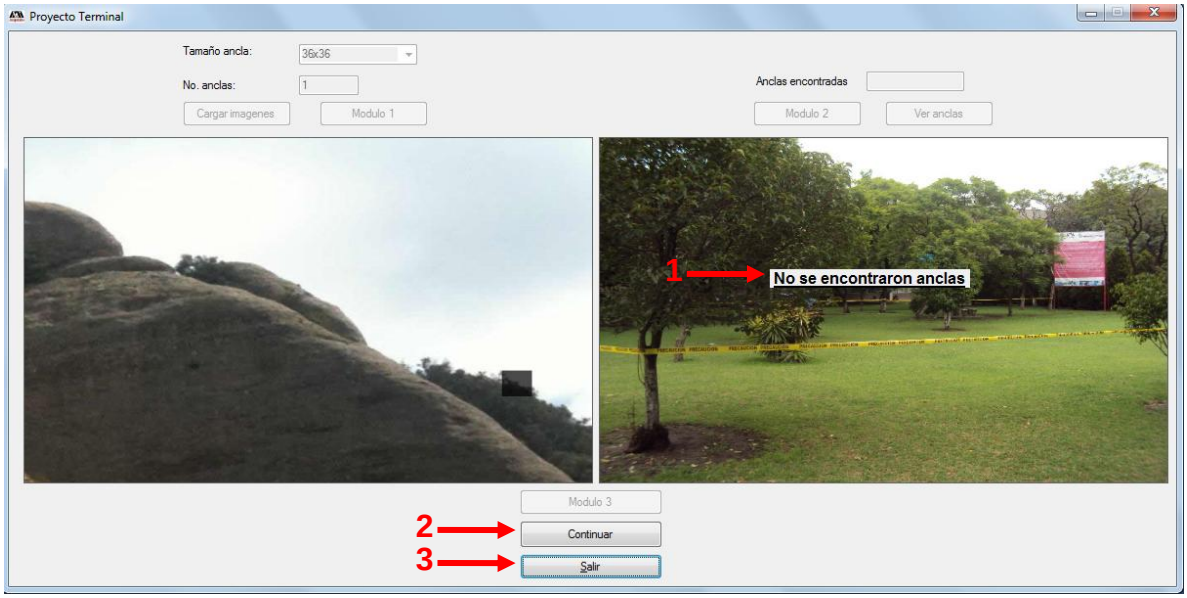
1. Caja para elegir el tamaño del ancla.
2. Cuadro de texto para introducir el número de anclas
3. Botón que despliega el cuadro de dialogo Abrir para seleccionar las imágenes a ser acopladas.
4. Área de imagen 1.
5. Área de Imagen 2.
6. Botón para salir de la aplicación.

Pantalla 2: Muestra de mejores anclas



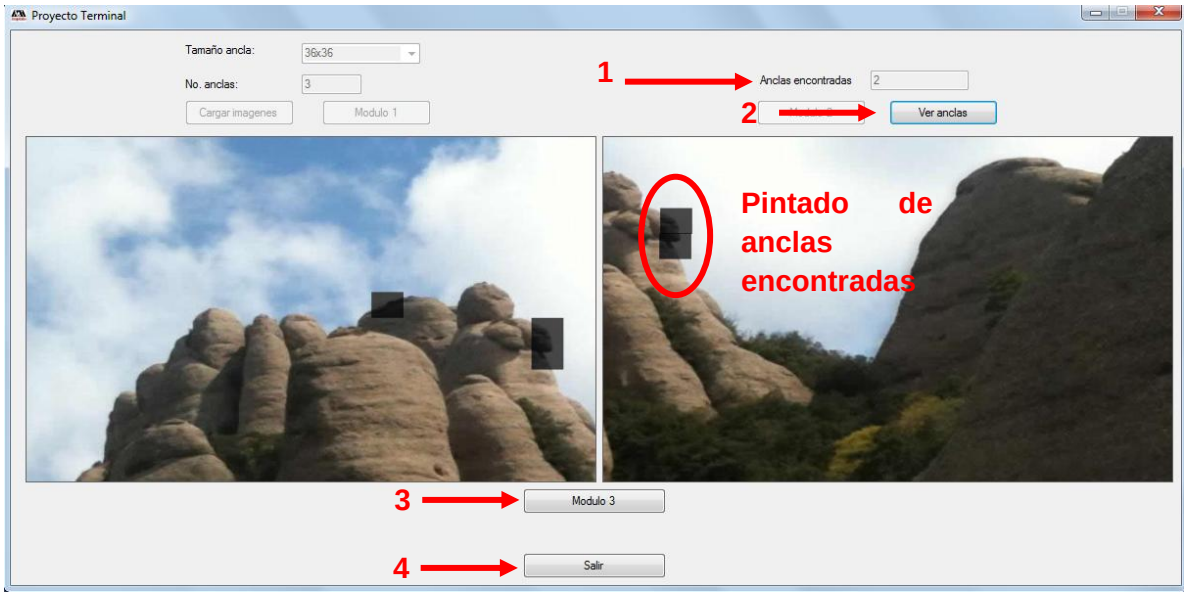
1. Botón para iniciar el proceso del módulo 2.
2. Mejores anclas encontradas
3. Botón para salir de la aplicación.

Pantalla 3: Módulo 2 no hay ancla



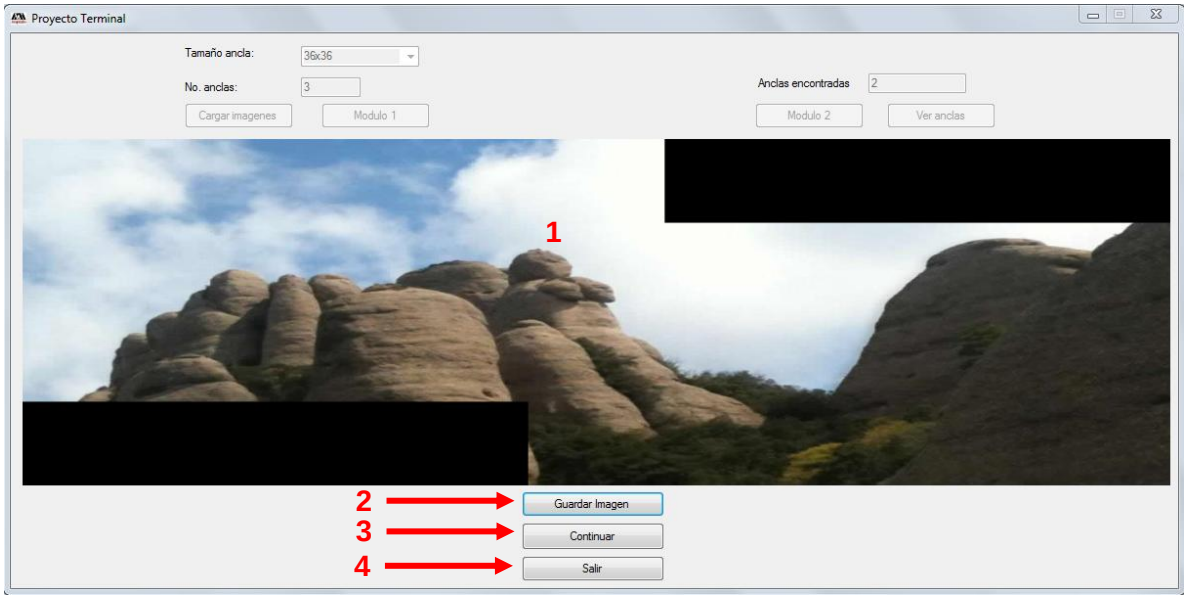
1. Mensaje “no se encontraron anclas.
2. Botón para continuar el proceso o para ir a pantalla principal.
3. Botón para salir de la aplicación.

Pantalla 4: Módulo 2 si hay anclas



1. Cuadro de texto que muestra en número de anclas encontradas
2. Botón para ver las anclas con las cuales se van a combinar las imágenes
3. Botón para iniciar el proceso del Módulo 3.
4. Botón para salir de la aplicación.

Pantalla 5: Imagen acoplada Módulo 3.



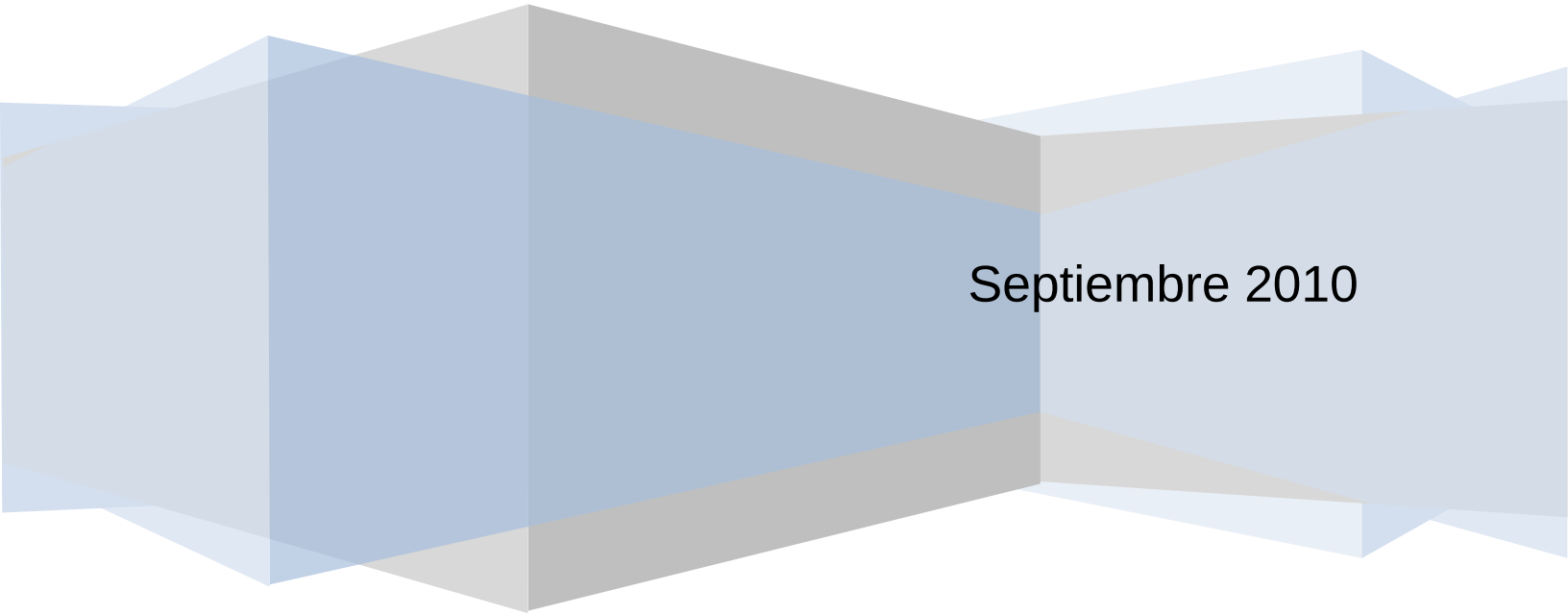
1. Área de imagen acoplada
2. Botón que abre cuadro de dialogo para guardar la imagen
3. Botón para continuar el proceso sin guardar la imagen
4. Botón para salir de la aplicación.

Universidad Autonoma Metropolitana
Unidad Azcapotzalco

Aplicación para Acoplamiento Automático por Similitudes de Imágenes Digitales

Manual de Usuario

Miguel Ángel Rojas Sandoval



Septiembre 2010

Contenido

1. Introducción

En este documento se describe los objetivos e información clara y concisa de cómo utilizar la aplicación de Proyecto Terminal de Acoplamiento Automático por Similitudes de Imágenes Digitales.

Este proyecto fué creado con el objetivo de diseñar e implementar un algoritmo para el acoplamiento de imágenes digitales a través del procesamiento digital de imágenes.

Es de mucha importancia consultar este manual antes y/o durante la utilización de la aplicación, ya que lo guiará paso a paso en el manejo de las funciones en él.

Con el fin de facilitar la comprensión del manual, se incluyen gráficos explicativos.

2. Objetivo de este manual

El objetivo principal de éste manual es ayudar y guiar al usuario a utilizar la aplicación de Proyecto Terminal de Acoplamiento Automático por Similitudes de Imágenes Digitales y comprende:

- Guía de instalación
- Guía para acceder a la aplicación.
- Conocer cómo utilizar la aplicación, mediante una descripción detallada e ilustrada de las opciones.

3. Requerimientos

3.1.1 Hardware

- Procesador: Mínimo 600 Mhz, recomendado 1 Ghz
- RAM: Mínimo 256, recomendado 1 GB
- HD: mínimo 3GB, recomendado 4.8 GB

3.1.2 Software

- Sistemas operativos Windows XP/Vista/7
- .NET Framework 3.5
- Windows Installer 3.1

4. Instalación

Antes de la instalación se debe contar con el software .Net Framework 3.5 para el funcionamiento de la aplicación y con Windows Installer 3.1 para ejecutar el instalador de la aplicación si no se cuenta con este software al iniciar la instalación el asistente nos ayudará a instalarlos a través de internet.

Una vez que se tengan todos los requerimientos tanto de software como de hardware para instalar la aplicación damos doble clic en cualquiera de los dos archivos que se tienen para la instalación el Setup.exe o el SetupPT.msi.

Figura 1. Instaladores

Se desplegará un asistente para la instalación de la aplicación como se muestra en la Figura 2.

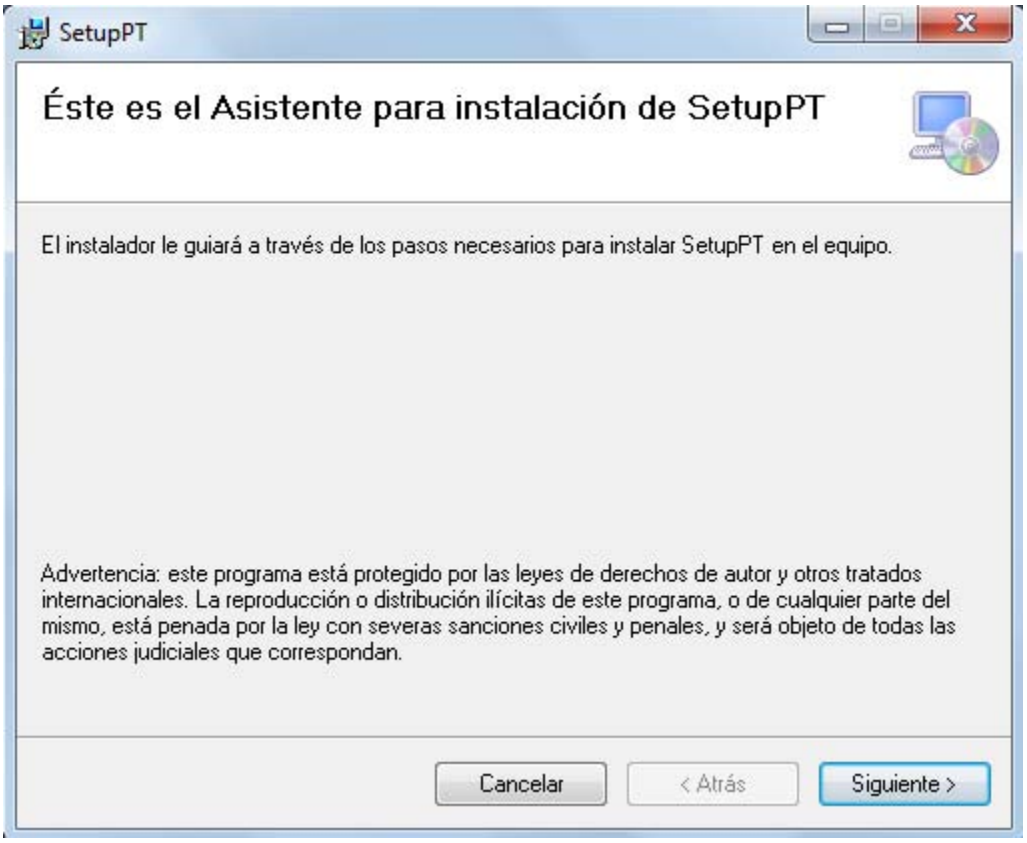


Figura 2: Asistente

Damos clic izquierdo en siguiente.

A continuación se elige el directorio en donde se instalará la aplicación, se recomienda dejar el que da el asistente por default pero si se desea cambiar se selecciona la opción examinar y damos clic en siguiente.

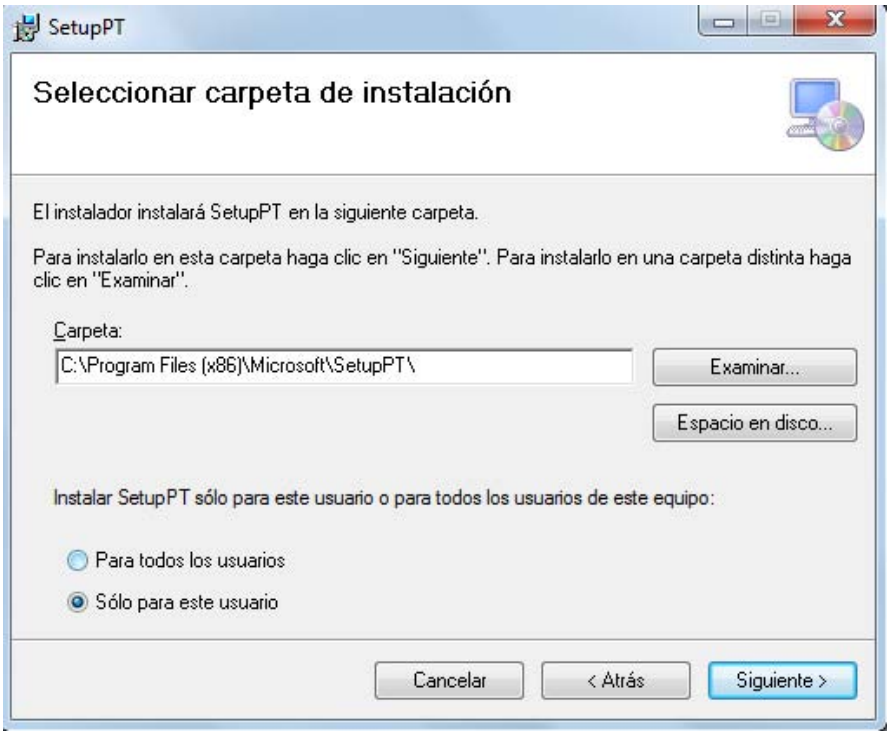


Figura 3. Asistente

Por último nos despliega una pantalla como la que se muestra a continuación y se da siguiente para que se inicie la instalación.

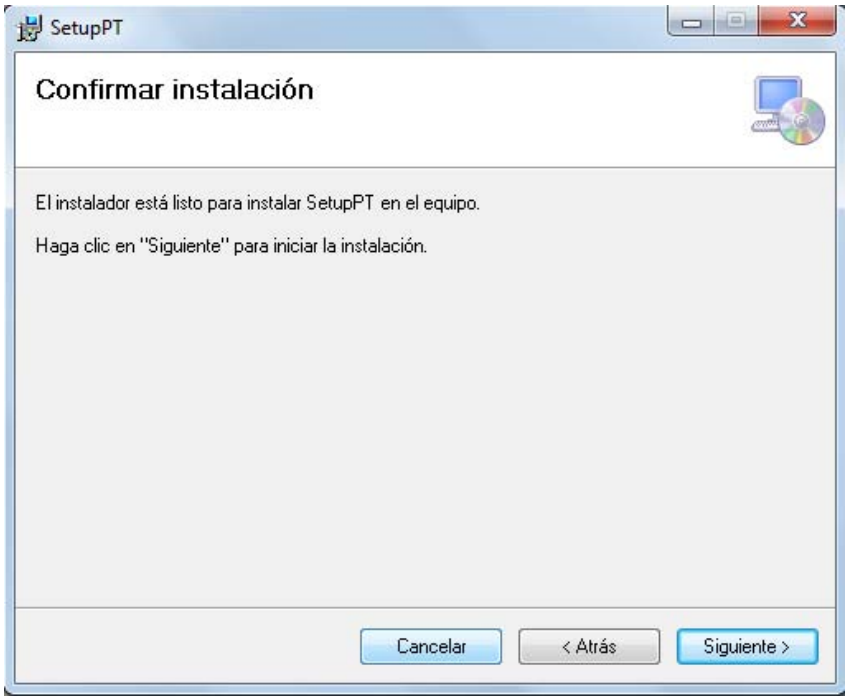


Figura 4. Asistente

Se despliega otra pantalla en donde se ve el progreso de la instalación como la que muestra a continuación.

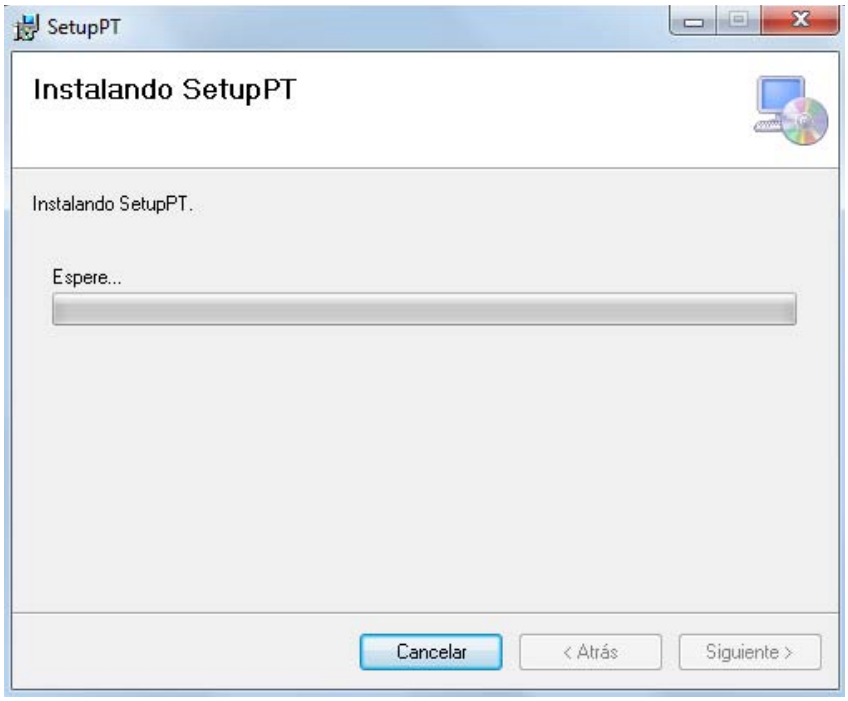


Figura 5. Instalando

Y finalmente se muestra un pantalla si no hubo ningún error en donde nos dice que la instalación se completo correctamente y damos clic en cerrar y listo la aplicación ya está instalada en tu ordenador.

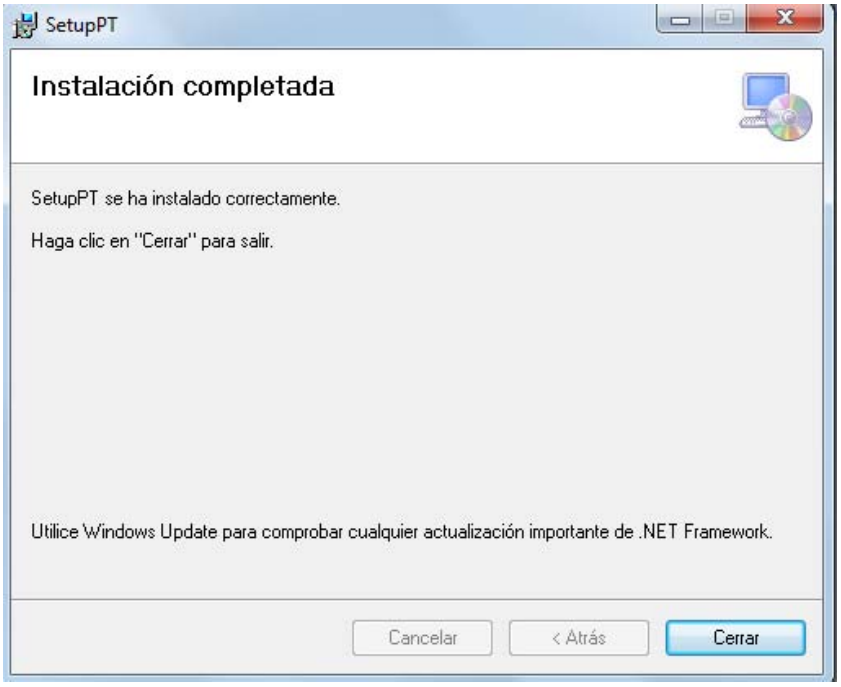


Figura 6. Instalación completa

5. Abrir aplicación

Para abrir la aplicación se tienen dos opciones.

- Abrir aplicación desde menú Inicio

Nos vamos a Menú Inicio- Todos los programas- Proyecto Terminal y seleccionamos PTv1.0.

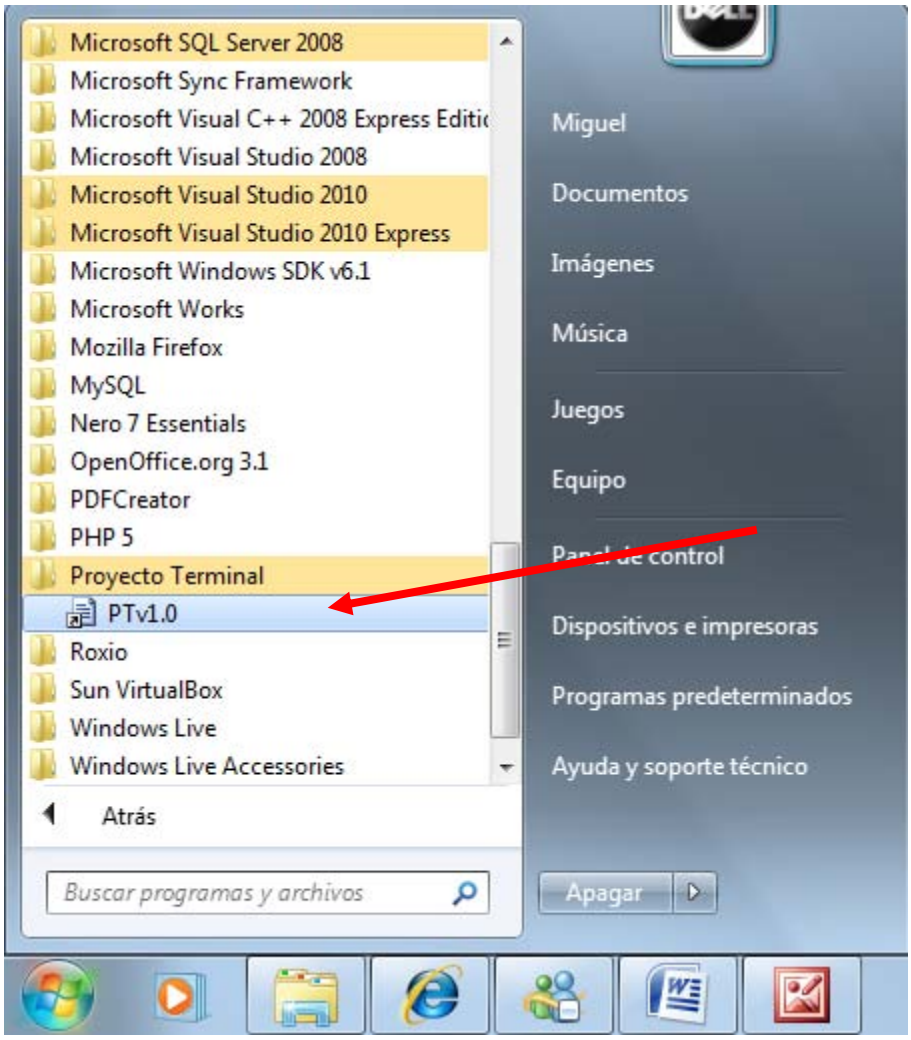


Figura 7. Inicio

- Abrir aplicación desde icono en escritorio

En el escritorio de Windows damos doble clic en el icono con nombre PTv.01 que se generó automáticamente al instalar la aplicación.

Figura 8. Icono

6. Aplicación

Una vez abierta la aplicación mostrará la siguiente pantalla:

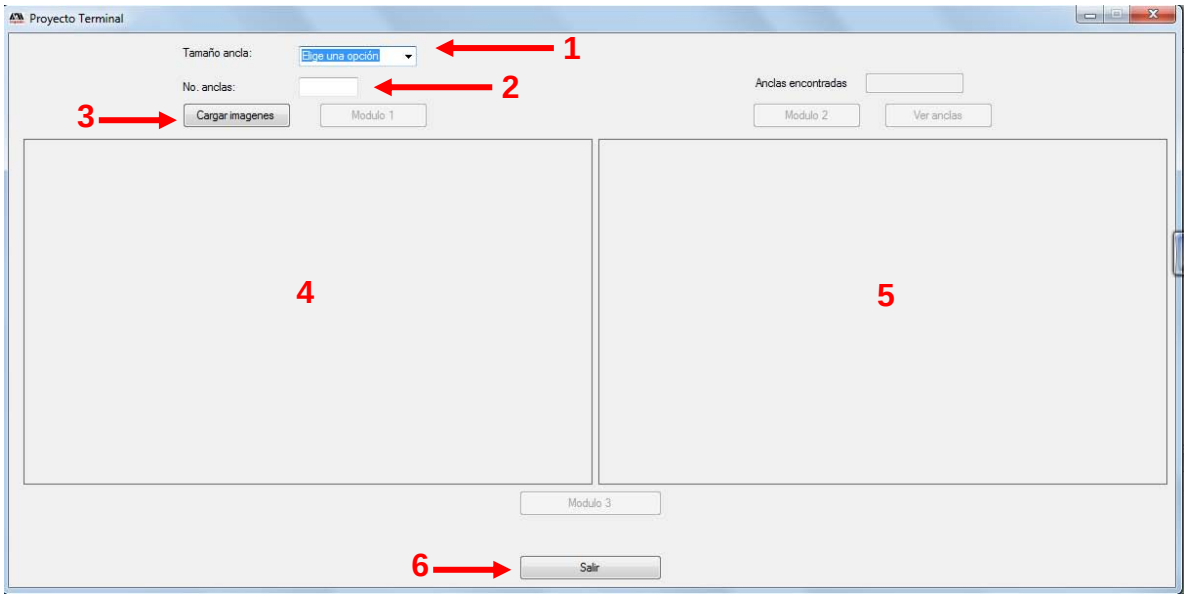


Figura 9. Cargar Imágenes

Esta pantalla iniciara con las siguientes opciones habilitadas:

1. Se especifica el tamaño del ancla dando clic en la flecha negra y se elije entre las opciones 9x9, 18x18, 27x27, 36x36

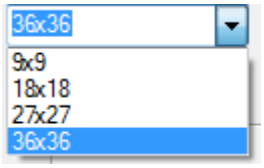


Figura 10. Tamaño del Ancla

2. Se introduce el número de anclas que se quiere que se busquen en las imágenes escribiéndolo en el cuadro de texto

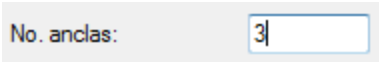


Figura 11. Número de anclas

3. Botón para cargar las imágenes

Figura 12. Botón Cargar Imágenes

4. Área para ver la imagen en donde se buscan las n mejores anclas.
5. Área para ver la imagen con las anclas que coinciden con la imagen del punto
6. Botón para salir de la aplicación.

Figura 13. Botón salir

Para empezar con el acoplamiento de imágenes antes de elegir las imágenes se tiene que ya haber elegido el tamaño del ancla y el número de anclas. Una vez elegidas estas opciones damos clic al botón cargar imágenes y nos aparecerá la siguiente pantalla.

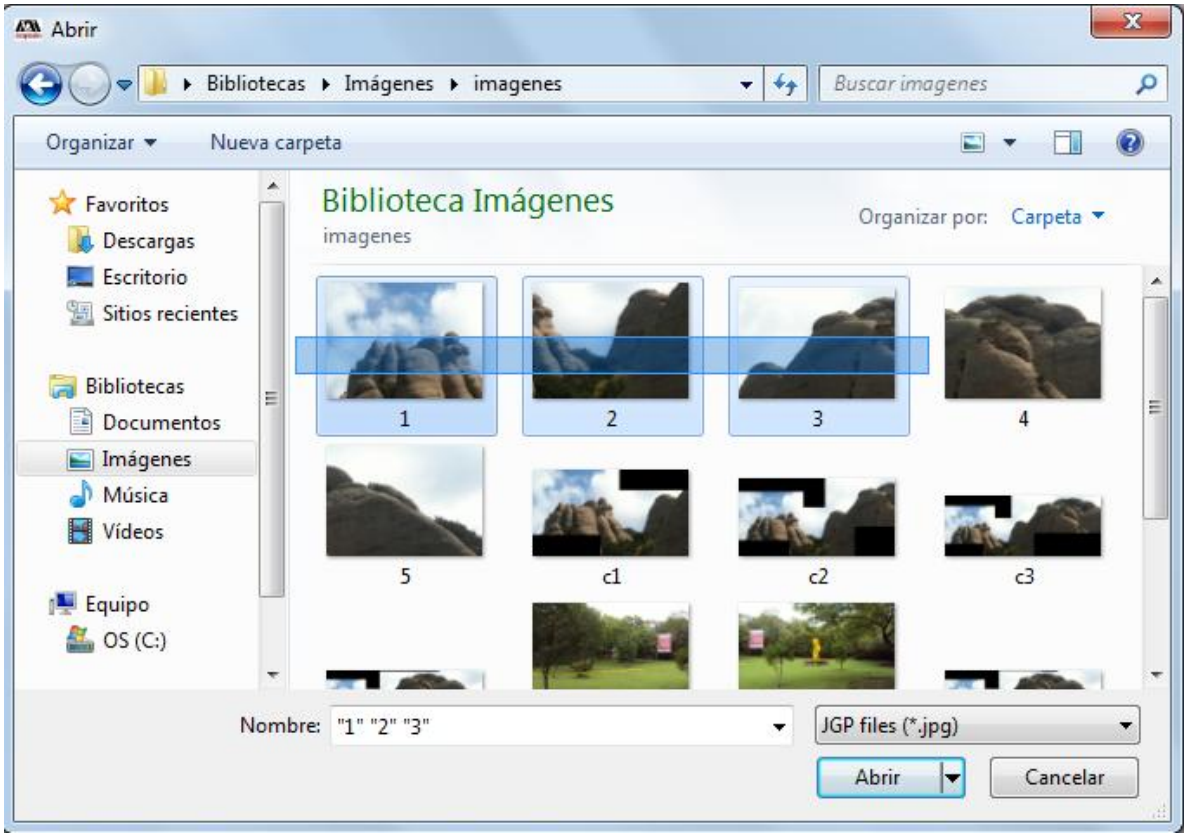


Figura 14. Cuadro de dialogo abrir

Al desplegarse este cuadro de dialogo navegamos hasta el directorio en donde se encuentran las imágenes y las seleccionamos en orden de acoplamiento de imágenes dando clic izquierdo y arrastrando hasta seleccionar todas las imágenes que deseamos, y por ultimo damos clic en el botón abrir.

Una vez elegidas las imágenes en la pantalla se habilitará el botón Modulo1 en la pantalla y damos clic izquierdo en este botón.

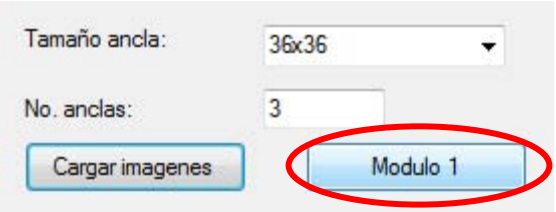


Figura 15. Inicio Módulo 1

Al ejecutar el botón Módulo 1 se inhabilitarán las cajas de texto tamaño del ancla y número de anclas y los botones cargar imágenes y Módulo 1, nos mostrará la imagen con las anclas encontradas con tamaño y número de anclas elegidas por el usuario con un color negro transparente y finalmente se habilitará el botón Módulo 2 tal y como muestra la Figura 16.

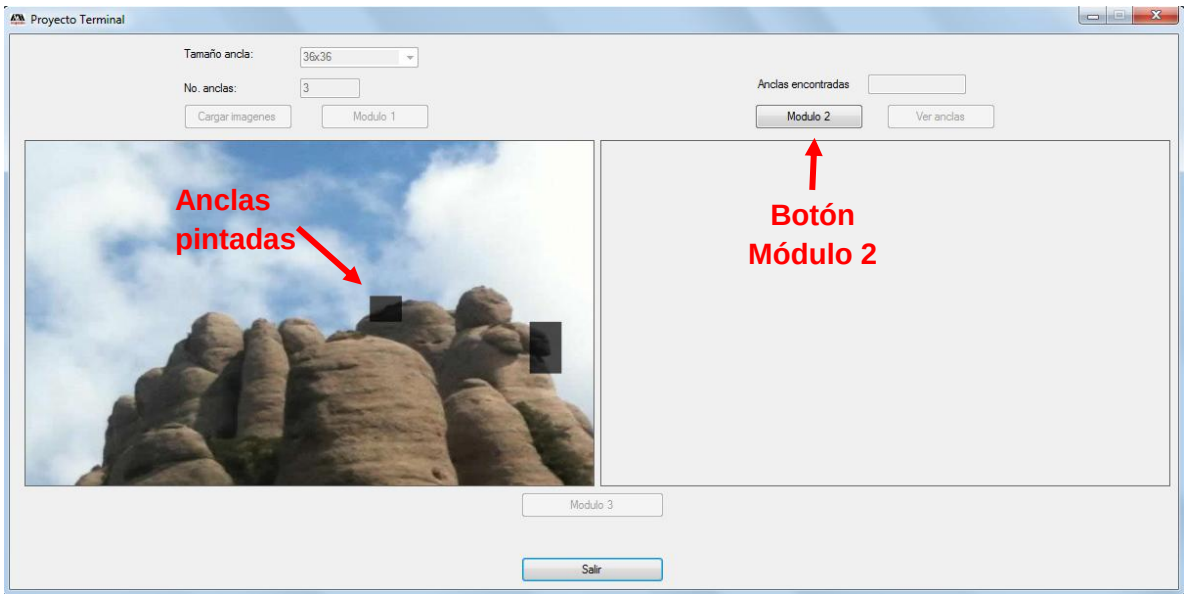


Figura 16. Mejores anclas

Al dar clic en el botón Módulo 2 tendremos dos opciones:

1. No se encontraron anclas
2. Se encontraron anclas

1. Si no se encontraron anclas aparecerá un mensaje en el centro de la imagen indicando que “No se encontraron anclas” y aparecerán los botones continuar el cual seguirá el proceso de acoplamiento si no se han analizado todas las imágenes ó terminara el proceso y mostrará la pantalla principal.

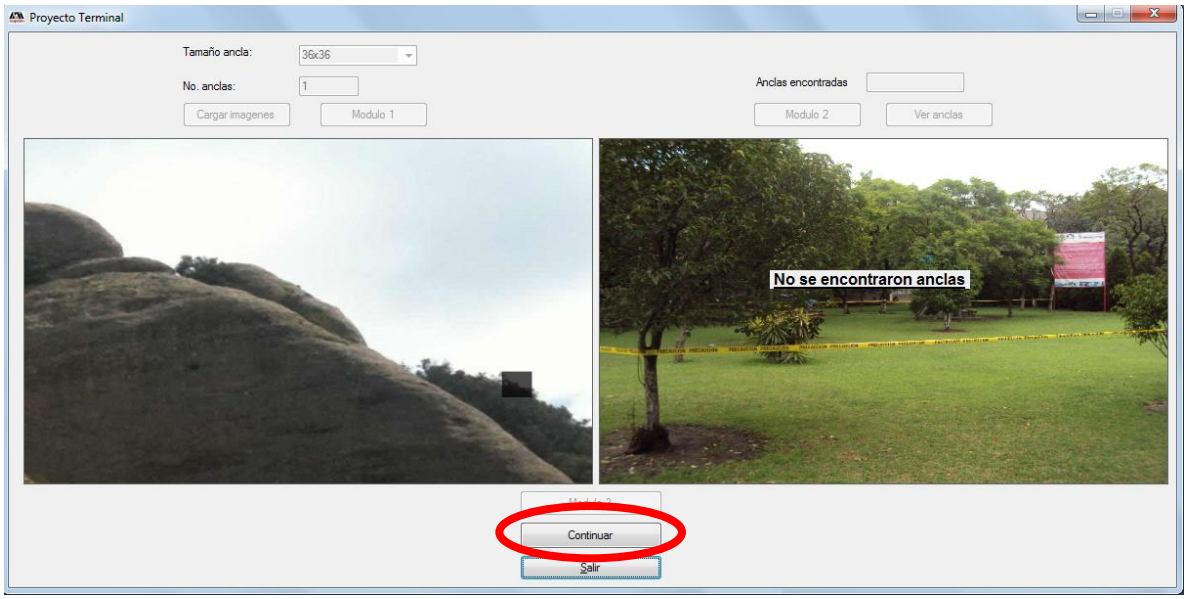


Figura 17. No se encontraron anclas

2. Si se encontraron anclas aparecerá la imagen con las anclas encontradas pintadas en la imagen con un color negro transparente, se deshabilitará el botón Módulo 2, aparecerá el número de anclas encontradas en el cuadro de texto anclas encontradas y se habilitarán los botones Ver anclas y Módulo 3 como muestra la Figura 18.

Figura 18. Anclas encontradas

Al dar clic el botón Ver anclas en los dos cuadros aparecerán las imágenes con la mejor ancla y más parecida entre las dos imágenes, a partir de la cual se unirán las dos imágenes y se deshabilitará el botón Ver anclas, como muestra la Figura 19.

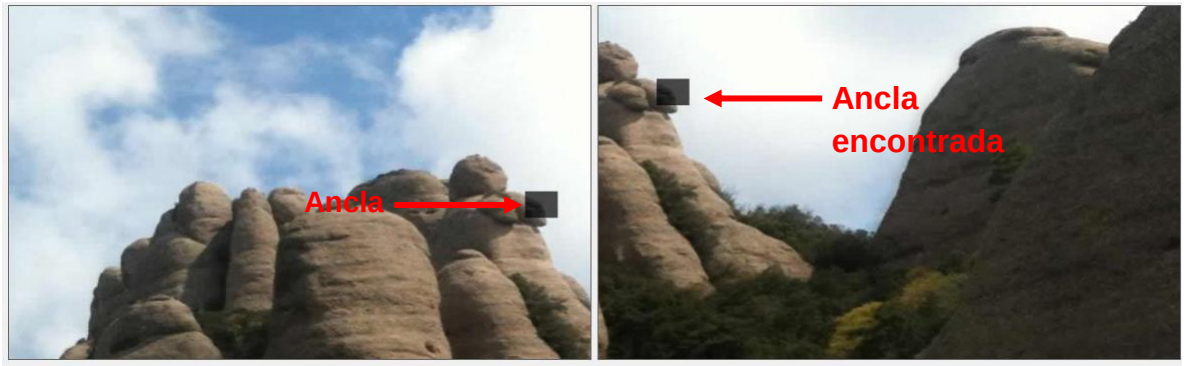


Figura 19. Ver anclas

Al dar clic izquierdo en el botón Módulo 3 aparecerá la imagen acoplada resultado de las dos imágenes a partir del ancla obtenida, se inhabilitarán los botones Guardar Imagen y Continuar y se deshabilitará el botón Modulo 3 como muestra la Figura 20.

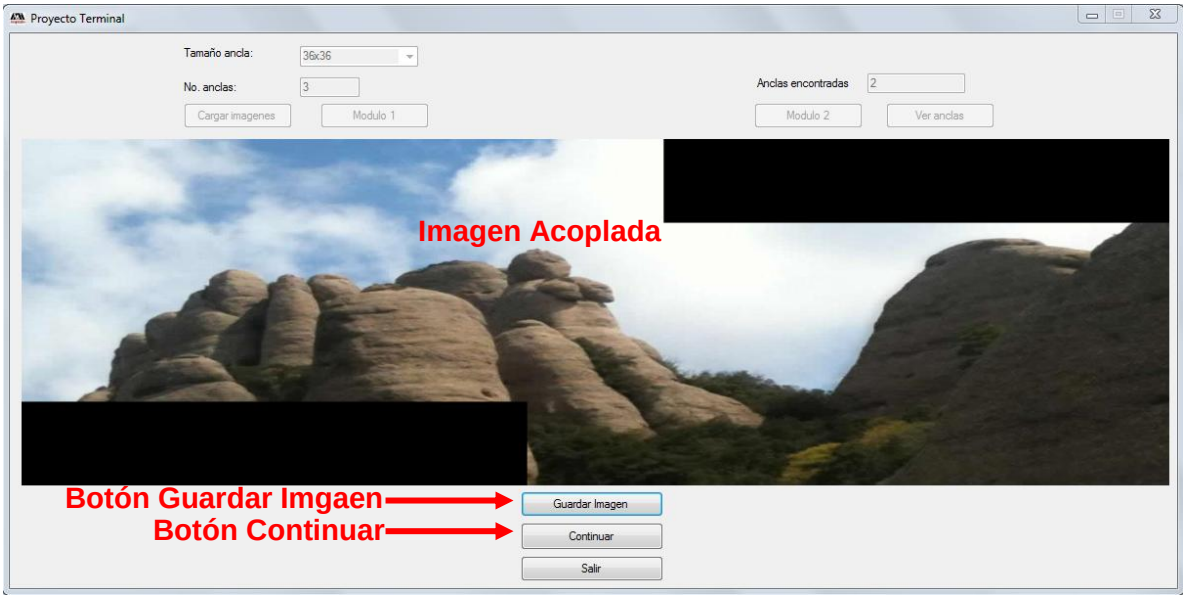


Figura 20. Imagen acoplada

Si damos clic izquierdo en el botón Guardar Imagen se desplegará un cuadro de dialogo en donde seleccionamos el lugar en donde se guardará la imagen acoplada e introducimos el nombre que se le asignará a la imagen. Al dar Guardar si ya se analizaron todas las imágenes la pantalla regresa a su estado inicial para analizar nuevas imágenes y si no se analizaron todas las imágenes, se repetirá el proceso desde el módulo 1 hasta analizar todas las imágenes.

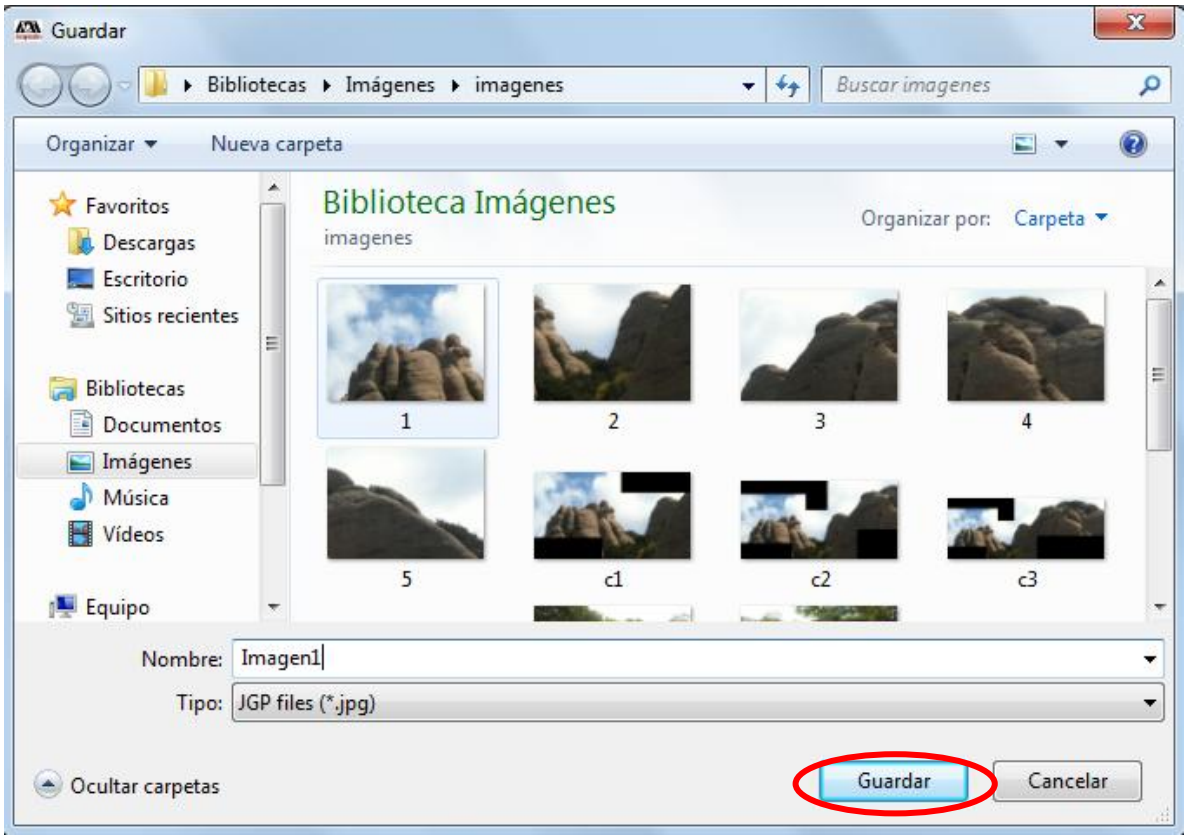


Figura 21. Cuadro de dialogo Guardar

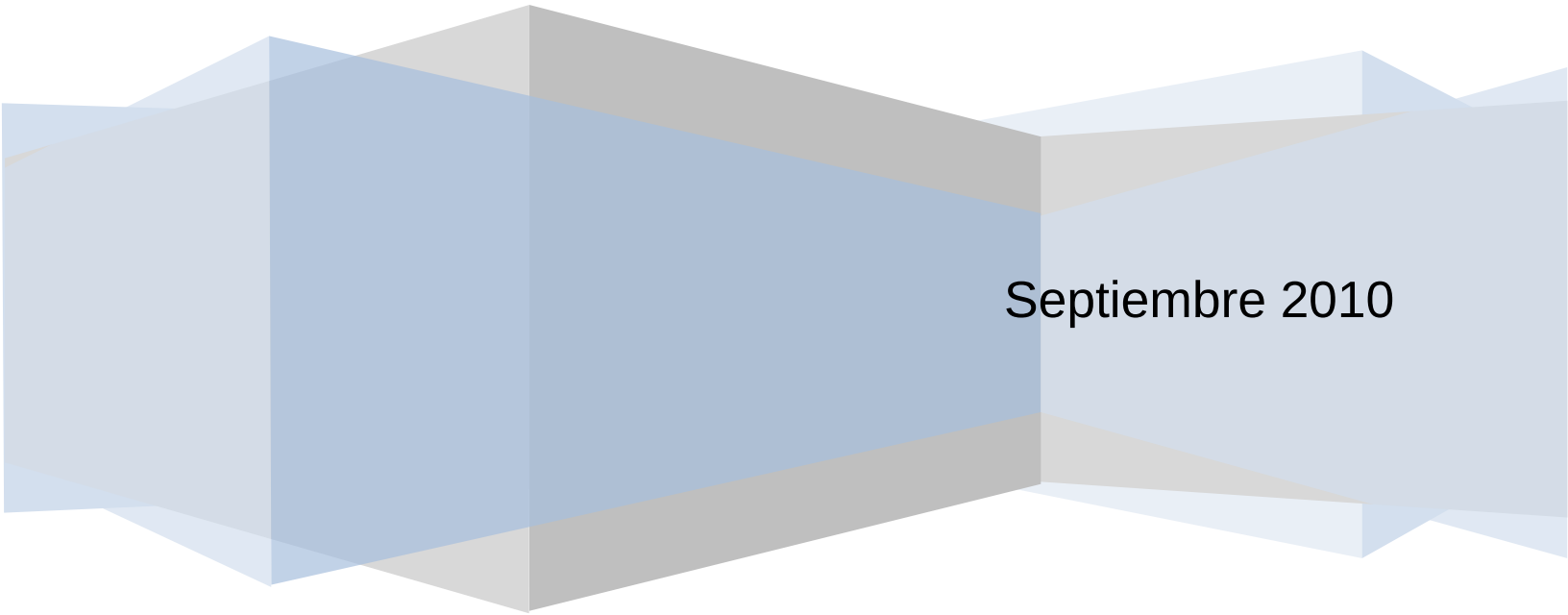
Si se selecciona el botón continuar no se guardará la imagen, y si ya se analizaron todas las imágenes la pantalla regresa a su estado inicial para analizar nuevas imágenes y si no se analizaron todas las imágenes, se repetirá el proceso desde el módulo 1 hasta analizar todas las imágenes.

Universidad Autonoma Metropolitana
Unidad Azcapotzalco

Aplicación para Acoplamiento Automático por Similitudes de Imágenes Digitales

Manual del programador

Miguel Ángel Rojas Sandoval



Septiembre 2010

Tabla de contenido

1. Introducción

En este documento se describen las clases que se han identificado en la fase del diseño. Se tratará de realizar una descripción, ya vista en el diseño, pero ahora adoptándola a la tecnología C#.

Se dará una descripción detallada de cada una de las clases, tanto como de sus atributos y métodos que las integran.

Este documento fué elaborado para ofrecer a los futuros programadores que quieran modificar o complementar esta aplicación, mostrar una visión sobre los procedimientos que se desarrollaron para lograr que esta aplicación cumpliera con cada uno de sus objetivos especificados.

En la primera sección del documento se muestra los módulos que lo comprenden y una breve descripción de sus datos de entrada y datos de salida.

En la segunda sección se muestra el diagrama de clases, de las distintas clases que comprende la aplicación.

Y por ultimo en la Tercera sección se describen cada una de las clases que involucran la aplicación de Proyecto Terminal para el Acoplamiento Automático por similitudes de Imágenes Digitales.

Esta aplicación fue realizada con el IDE Microsoft Visual C# 2008 Express Edition con .Net Framework 3.5.

2. Objetivo del documento

El objetivo del documento consiste en brindar al lector una visión sobre la arquitectura de la aplicación de Proyecto Terminal para el Acoplamiento Automático por similitudes de Imágenes Digitales.

3. Módulos de la aplicación

Este proyecto está compuesto por 3 módulos (diagrama 1), el *módulo de generación de descriptores, módulo buscador en siguiente imagen y el módulo combinador de imagen.* Para cumplir el objetivo principal es necesario el correcto funcionamiento de todos los módulos.

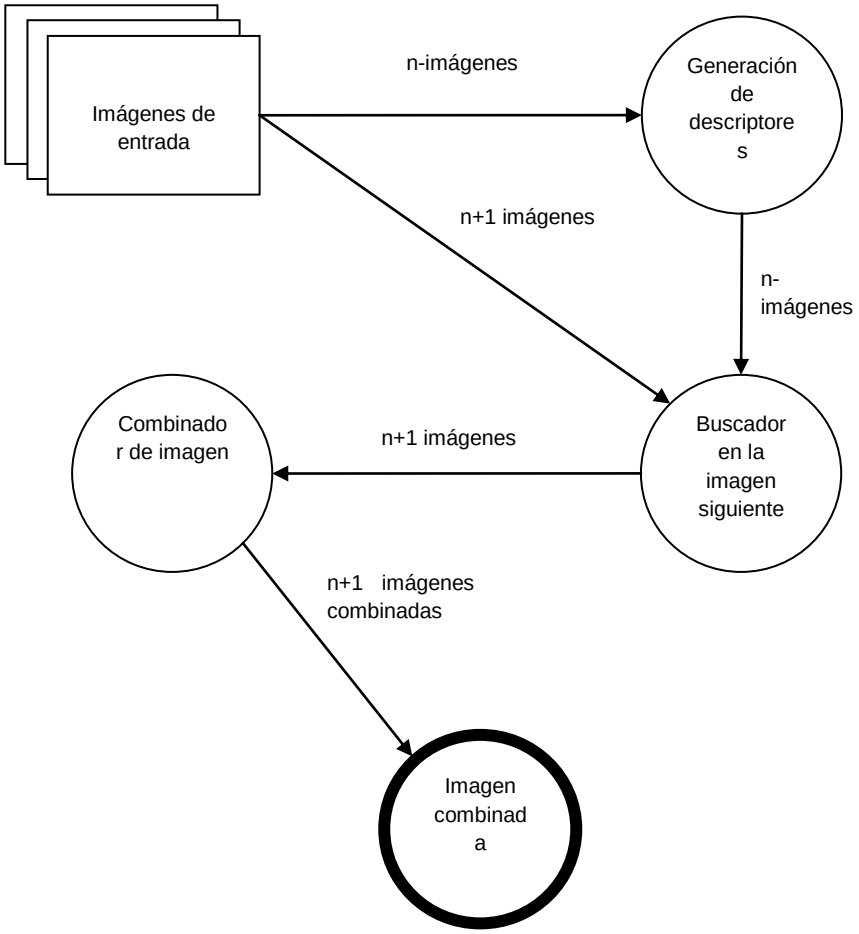


Figura 1. Módulos para acoplamiento de imágenes horizontales

Módulo de generación de descriptores: Este módulo básicamente se subdivide en dos etapas, la primera etapa se encarga de que dada una imagen, calcula su equivalente de la imagen en escala de grises para poder obtener un histograma en base a sus niveles de gris ya que gracias a éste se obtienen los descriptores estocásticos, une vez obtenida la imagen en escala de grises, en la etapa dos se analiza la imagen en un 40% aproximadamente de su extremo derecho por segmentos de matrices de pixeles, cuyas dimensiones serán datos de entrada, de cada una de estas matrices de pixeles se obtendrá un vector con una serie de descriptores estocásticos, dichos descriptores serán comparados con los de otras matrices de pixeles con el objetivo de saber cuáles van a ser considerados como anclas.

Módulo buscador de la imagen siguiente: Este módulo se encargará de comparar las mejores m anclas obtenidas en el módulo anterior, con todas las posibles regiones de pixeles que se puedan obtener de la siguiente imagen, evaluándolas en el 40% del extremo izquierdo de la imagen a ser acoplada. Esta comparación con las anclas se hará en base a los descriptores estocásticos tanto de la imagen 1 como de la imagen 2 para buscar las que son iguales y puedan acoplarse. De las anclas de la imagen 1 y de la imagen 2 que sean semejantes se obtendrán sus coordenadas.

Módulo combinador de imagen: El objetivo de este módulo será acoplar, en base a las coordenadas de las anclas obtenidas en el módulo anterior, las dos imágenes.

Una vez que el último módulo cumpla su objetivo, el proceso se volverá a repetir en dado caso de que se tengan más imágenes para ser acopladas, en caso contrario terminará el proceso.

A continuación en el Diagrama 2 se muestran los datos de entrada y los datos de salida que involucran cada módulo.

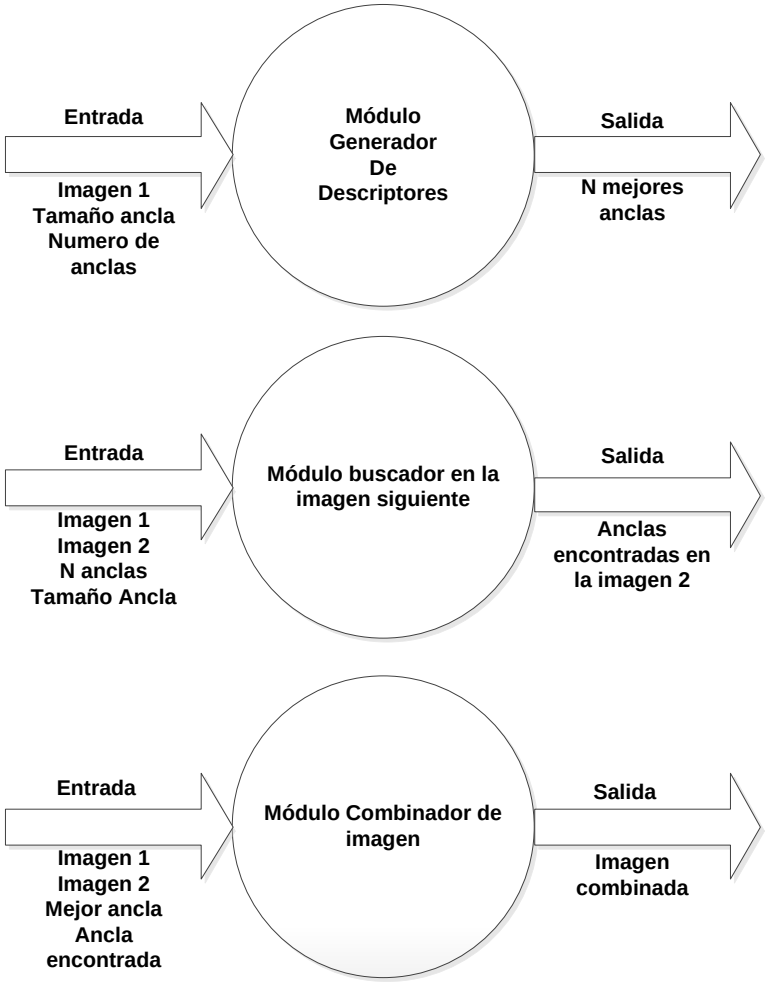


Figura 2. Diagrama de estados 2

4. Diagrama de clases

En el diagrama de la figura 3 se muestran las clases que se obtuvieron al desarrollar la aplicación.

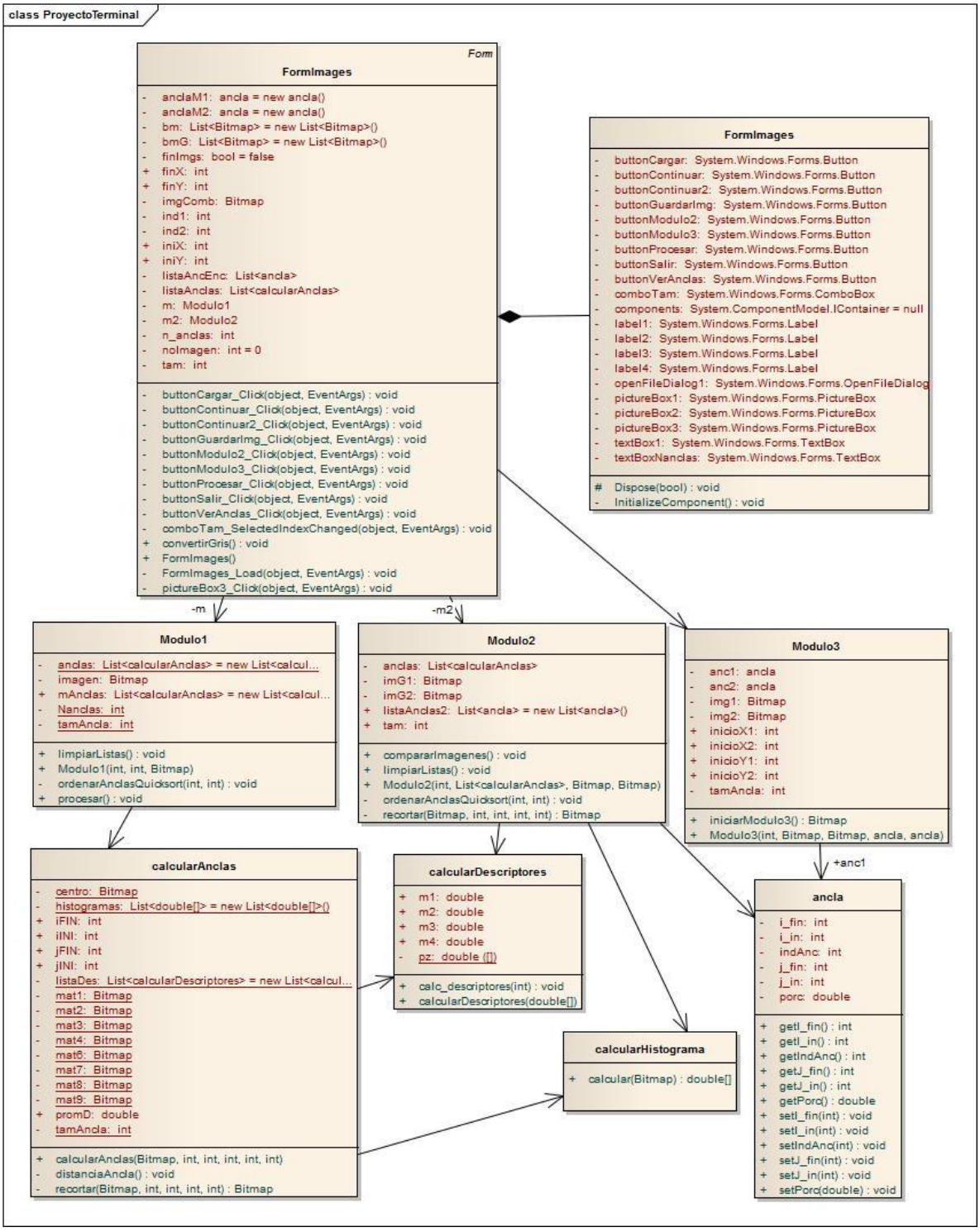


Figura 3. Diagrama de clases

La clase FormImages es la clase principal de esta aplicación a partir de esta clase se llaman a las clases del Módulo 1, Módulo 2 y Módulo 3.

El siguiente diagrama muestra las clases separadas en paquetes involucradas en cada módulo

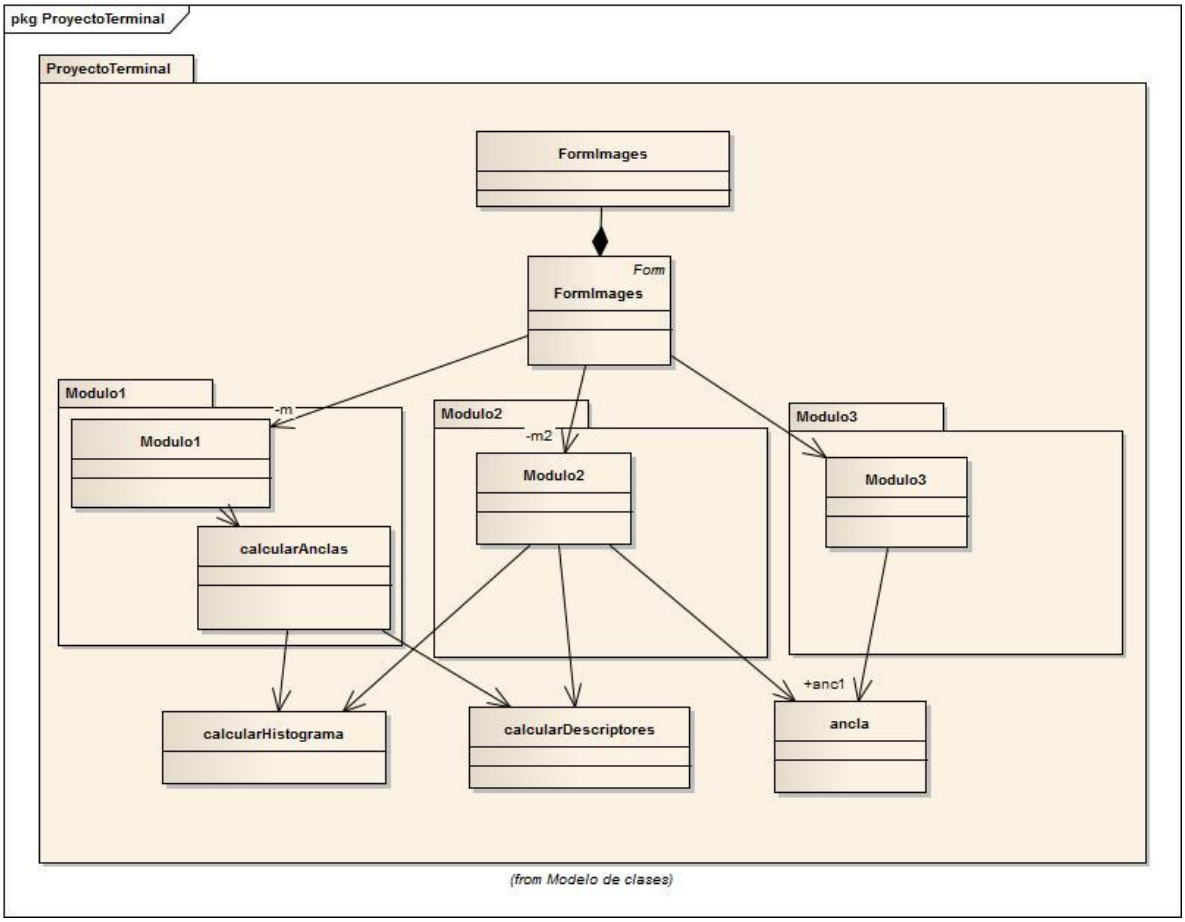


Figura 4. Diagrama en paquetes

5. Clases

5.1 Clase FormImages

Clase principal de la aplicación y del formulario de la aplicación. A partir de esta clase se dan los datos de entrada y se llama a cada uno de los módulos de la aplicación.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
bm	List<Bitmap>	Private	Lista para almacenar las imagenes
bmG	List<Bitmap>	Private	Lista para almacenar imágenes en escala de grises
listaAnclas	List<calcularAnclas>	Private	Lista para almacenar las mejores anclas

listaAncEnc	List<ancla>	Private	Lista para almacenar las enclas encontradas
imgComb	Bitmap	Private	Bitmap para imagen final de modulo 3
m	Modulo 1	Private	Instancia de clase Módulo 1
m2	Modulo 2	Private	Instancia de clase Modulo 2
tam	int	Private	Tamaño del ancla
noImagen	int	Private	Índice de la imagen que se analiza
n_anclas	int	Private	Dato de entrada de número de anclas
anclaM1	ancla	private	Intancia de la clase ancla para el ancla a acoplar de imagen 1.
anclaM2	ancla	private	Intancia de la clase ancla para el ancla a acoplar de imagen 2.
iniX	int	public	Índice x del inicio en donde se acopla la imagen en la imagen final combinada
iniY	int	public	Índice Y del inicio en donde se acopla la imagen en la imagen final combinada
finX	int	public	Índice x del final en donde se acopla la imagen en la imagen final combinada
finY	int	public	Índice Y del final en donde se acopla la imagen en la imagen final combinada
finImgs	bool	private	Atributo para saber si se están analizando el último par de imágenes

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`buttonCargar_Click(object sender, EventArgs e)`	void	private	Evento del botón cargar, carga las imágenes en bm
`buttonProcesar_Click(object sender, EventArgs e)`	void	private	Evento del botón procesar, inicializa Modulo 1
`convertirGris()`	void	public	Método para convertir las imágenes en escala de grises
`buttonModulo2_Click(object sender, EventArgs e)`	void	private	Evento del botón Modulo 2, inicializa módulo 2.
`buttonModulo3_Click(object sender, EventArgs e)`	void	private	Evento del botón Modulo 3, inicializa Módulo 3.
`buttonVerAnclas_Click(object`	void	Private	Evento del botón Ver

sender, EventArgs e)			anclas pinta las anclas que se van acoplar
buttonGuardarImg_Click(object sender, EventArgs e)	void	private	Evento del botón GuardarImg, cuadro de dialogo para guardar imagen.
buttonContinuar_Click(object sender, EventArgs e)	void	private	Evento del botón Continuar, continuar para la siguiente imagen
buttonContinuar2_Click(object sender, EventArgs e)	void	private	Evento del botón continuar2, manda a pantalla principal
buttonSalir_Click(object sender, EventArgs e)	void	private	Salir de la aplicación

5.2 Clase Módulo 1

Esta clase encarga de seguir paso a paso las etapas del módulo 1 primero toma la imagen, calcula el 40% del extremo derecho de la imagen y lo analiza por segmentos de pixeles del tamaño del ancla pixel a pixel e invoca a la clase calcular anclas para obtener sus descriptores estocásticos y finalmente elige las mejores anclas.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
tamAncla	int	static	Parámetro correspondiente al tamaño del ancla
Nanclas	int	static	Parámetro correspondiente al numero de anclas a calcular
imagen	Bitmap	private	Imagen para calcular las anclas
mAnclas	List<calcularAnclas>	public	Lista con las mejores anclas
anclas	List<calcularAnclas>	static	Lista con todos los segmentos con sus descriptores

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
procesar()	void	public	Controla los procesos del modulo 1, llama a caluclar anclas y obtiene los descriptores de cada segmento
limpiarListas()	void	Public	Limpia las listas
ordenarAnclasQuicksort(int primero, int ultimo)	void	public	Ordena la lista de ancas a parrir del promD de cada ancla

5.3 Clase calcularAnclas

A esta clase le mandan las dimensiones del segmento recorte el pedazo de la imagen en Gris y las divide en 9 segmentos calcula el histograma y los descriptores de cada uno de los 9 segmentos, saca la distancia lógica del segmento del centro con los de los lados y obtiene el promedio de todos.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
tamAncla	int	static	Parámetro del tamaño del ancla
mat1	Bitmap	static	Segmento superior izquierdo
mat2	Bitmap	static	Segmento superior centro
mat3	Bitmap	static	Segmento superior derecho
mat4	Bitmap	static	Segmento izquierdo
centro	Bitmap	static	Segmento centro
mat6	Bitmap	static	Segmento derecho
mat7	Bitmap	static	Segmento inferior izquierdo
mat8	Bitmap	static	Segmento inferior centro
mat9	Bitmap	static	Segmento inferior derecho
histogramas	List<doublé[]>	static	Lista con los histogramas de los 9 segmentos
listaDes	List<calcularDescriptores>	static	Lista con los descriptores de cada segmento
promD	double	public	promedio del ancla
iINI	int	public	Índice i de donde inicia el segmento en la imagen
jINI	int	public	Índice j de donde inicia el segmento en la imagen
iFIN	int	public	Índice i de donde termina el segmento en la imagen
jFIN	int	public	Índice j de donde termina el segmento en la imagen

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`distanciaAncla()`	void	private	Obtiene las distancia lógicas de cada par de segmentos en base a la resta de sus descriptores y obtiene el promedio
`recortar(Bitmap imagen, int j_in, int i_in, int j_fin, int i_fin)`	Bitmap	private	Recorta los 9 segmentos de él ancla

5.4 Clase calcularHistograma

Esta clase calcula el histograma de una imagen o segmento de imagen enviada.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`calcular(Bitmap segImg)`	doublé[]	public	Calcula el histograma de una imagen en escala de grises

5.5 Clase calcularDescriptores

Esta clase calcula los descriptores de un histograma.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
pz	doublé[]	static	histograma
m1	double	public	Descriptor de grado 1
m2	double	public	Descriptor de grado 2
m3	double	public	Descriptor de grado 3
m4	double	public	Descriptor de grado 4

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`calcular(Bitmap segImg)`	doublé[]	public	Calcula el histograma de una imagen en escala de grises

5.6 Clase Modulo2

Esta clase se encarga de comparar las dos imágenes y busca en la imagen dos las anclas de la imagen 1 por segmentos de imágenes del tamaño del ancla pixel a pixel en el 40% del extremo izquierdo de la imagen2.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
tam	int	public	Tamaño del ancla
anclas	List<calcularAnclas>	private	Lista con anclas de la imagen 1
listaAnclas2	List<ancla>	public	Lista en donde se almacenan las anclas encontradas
imG1	Bitmap	prívate	Imagen 1
imG2	Bitmap	prívate	Imagen 2

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`compararImagenes()`	void	public	Recorta y calcula los descriptores de la imagen 2 para compararlos con los de la imagen 1.
`ordenarAnclasQuicksort(int primero, int ultimo)`	void	private	ordena las anclas encontradas a partir de su porcentaje
`recortar(Bitmap imagen, int j_in, int i_in, int j_fin, int i_fin)`	Bitmap	private	Recorta un segmento de la imagen
`limpiarListas()`	void	public	Limpia las listas de la clase

5.7 Clase ancla

Clase de tipo *get* y *set* para almacenar los campos de cada ancla.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
i_in	int	private	Índice i de inicio de el ancla
i_fin	int	private	Índice i de fin de el ancla
j_in	int	private	Índice j de inicio de el ancla
j_fin	int	private	Índice j de fin de el ancla
porc	double	private	Porcentaje de diferencia entre las anclas

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`getI_in()`	int	public	Método get de atributo i_in
`getI_fin()`	int	public	Método get de atributo i_fin
`getJ_in()`	int	public	Método get de atributo j_in
`getJ_fin()`	int	public	Método get de atributo j_fin
`getIndAnc()`	int	public	Método get de atributo indAnc
`getPorc()`	int	public	Método get de atributo porc
`setI_in()`	void	public	Método set de atributo i_in
`setI_fin()`	void	public	Método set de atributo i_fin
`setJ_in()`	void	public	Método set de atributo j_in
`setJ_fin()`	void	public	Método set de atributo

			j_fin
`setIndAnc()`	void	public	Método set de atributo indAnc
`setPorc()`	void	public	Método set de atributo porc

5.8 Clase Modulo3

Clase que se encarga de acoplar las dos imágenes a partir de un ancla en común.

Atributos			
Nombre	**Tipo**	**Alcance**	**Descripción**
tamAncla	int	private	Tamaño del ancla
img1	Bitmap	private	Imagen 1
img2	Bitmap	private	Imagen 2
anc1	ancla	private	Instancia de la clase ancla
anc2	ancla	private	Instancia de la clase ancla
inicioX1	int	public	Posición de índice x en donde inicia ancla
inicioY1	int	public	Posición de índice y en donde inicia ancla
inicioX2	int	public	Posición de índice x en donde inicia ancla encontrada
inicioY2	int	public	Posición de índice y en donde inicia ancla encontrada

Operaciones			
Nombre	**Valor de retorno**	**Alcance**	**Descripción**
`iniciarModulo3()`	Bitmap	public	Combina las dos imágenes a partir de el ancla