Universidad Autónoma Metropolitana Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Separación automática de fondo y fenómeno de interés en imágenes tipo círculos en los cultivos

Luis Antio Valerio Gayosso
Matrícula: 208300414

Trimestre 2013 Primavera
21 de Agosto de 2013
Reporte final

Asesor
Risto Fermín Rangel Kuoppa
Profesor Titular
Departamento de Sistemas

Resumen

Conforme transcurre el tiempo el procesamiento digital de imágenes (PDI), se va desarrollando e incrementando su nivel y amplitud de conocimiento, así como técnicas y métodos para obtener resultados deseados a partir de imágenes, que por alguna razón, son de interés personal o social para su investigación u obtención de información.

En este proyecto se aplican algunas técnicas del PDI en secuencia para la obtención de información a partir de una imagen de entrada, dicha imagen pude ser cualquiera que cumpla las especificaciones del software, sin embargo, dicho software fue diseñado para trabajar con imágenes tipo 'figuras en cultivos', para las cuales al final se obtendrá como resultado el contorno de la figura formada en dicha imagen.

Como bien se mencionó ya, este proyecto está diseñado para trabajar con imágenes tipo 'figuras en cultivos', sin embargo si el usuario desea obtener un resultado similar para otro tipo de imagen este puede ser utilizado y obtener resultados excelentes.

Las etapas por las que pasa la imagen para ser procesada y modificada, son las siguientes: filtrado gaussiano, binarizado, segmentación y adelgazamiento. Dichas etapas son aplicadas en el mismo orden a la imagen en el software.

Es importante que en cada etapa se obtengan los resultados más apropiados para el objetivo buscado, de no ser así podría contribuir y generar algún error que se propague a través de las etapas de procesamiento de la imagen y al final obtener una imagen con información errónea o poco acercada al resultado deseado.

Tabla de contenido

Objetivos

Extraer el contorno de la región donde se da el fenómeno 'círculos en los cultivos' en imágenes de cosecha, por medio de técnicas de procesamiento digital de imágenes.

Objetivos específicos

- Quitar el ruido de alta frecuencia[1] de la imagen fuente.
- Identificar los píxeles[2] que forman el fenómeno de interés en la imagen.
- Agrupar los píxeles que coincidan con el fenómeno de interés en una o más áreas sobre la imagen original.
- Definir el contorno de las áreas con el fenómeno de interés.
- Convertir el contorno de las áreas a una enumeración de coordenadas en la imagen original.

Introducción

El procesamiento digital de imágenes (PDI) tiene como objetivo extraer la información de una imagen para hacerla manipulable en una computadora, para que después, por medio de técnicas y métodos, se mejore su calidad o se pueda extraer información de ésta para cumplir el objetivo que se requiera.

Aunque el PDI es relativamente reciente (década de los sesenta)[3] , ya se tiene un amplio campo de estudio y áreas donde es de gran ayuda para realizar tareas automatizadas o con mayor precisión, ejemplos de esto son la medicina, la inteligencia artificial, el diseño gráfico, la astronomía, la fotografía, la geografía, entre otras.

Para este proyecto nos basaremos sólo en la extracción de determinada información deseada, excluyendo la información que no interesa. Ejemplo de aplicaciones donde se requiere llegar a un resultado similar son: imágenes de ultrasonidos, detección de quistes, enfermedades oftalmológicas[4], visión por computadora, reconocimiento facial, análisis de huella digital, mejoramiento de calidad de imagen, reconocimiento óptico de caracteres, mapas geográficos, reconocimiento de gestos, etc.

Las imágenes con círculos en los cultivos, también conocidas como *"crop-circles"* en inglés, son aquellas en donde se puede observar una formación geométrica sobre el campo de cosecha (trigo, maíz, etc.), ver Figura1. La primera imagen con círculos en los cultivos de la que se tiene fecha fue en 1678 hecha en Inglaterra **[1]**.

[1] Ruido de alta frecuencia: información contenida en una imagen que destaca de la demás y que no es relevante para el objetivo que se desea realizar.
[2] Píxel: unidad mínima de composición de una imagen.
[3] http://design.osu.edu/carlson/history/lesson1.html.
[4] Oftalmología: parte de la patología que estudia las enfermedades de los ojos.

Sin embargo, a finales de la década de los setenta comenzaron a surgir nuevas formas, cada vez más complejas y definidas, lo que causó gran controversia de cuál era el origen y para qué se hacían estas formas en los cultivos.

Figura 1. Dibujo sobre un campo de cultivo.

Para este proyecto no será de interés ni se estudiarán cuáles son las causas o efectos de los círculos en los cultivos, ni se pretende saber quién, para qué y cómo se originan. El proyecto sólo se enfoca en el dibujo que se forma sobre la imagen.

Dado que el origen de *crop circles* aún no tiene explicación total, hay grupos de investigación (por ejemplo BLT[5]), dedicados al estudio de los posibles orígenes de este fenómeno. En este proyecto terminal se desarrollará una herramienta que apoyará al estudio de la estructura y patrones en las figuras que se forman en los cultivos, lo cual podría contribuir a esclarecer el origen de dicho fenómeno.

El proyecto básicamente es extraer a través de métodos y procesamiento digital, el contorno del dibujo que se forma en el campo de cultivo, descartando ruido y otras formas que no pertenezcan al dibujo.

Desarrollo del proyecto

Como ya se mencionó, el proyecto tiene varias etapas en donde se aplican las técnicas de PDI siguientes: filtrado gaussiano, binarizado, segmentación y adelgazamiento. Estas etapas son aplicadas respectivamente a la imagen.

El diagrama 1 muestra la interacción entre los módulos que procesarán la imagen de entrada.

[5] BLT research team, Inc. Página principal: http://www.bltresearch.com/history.php.

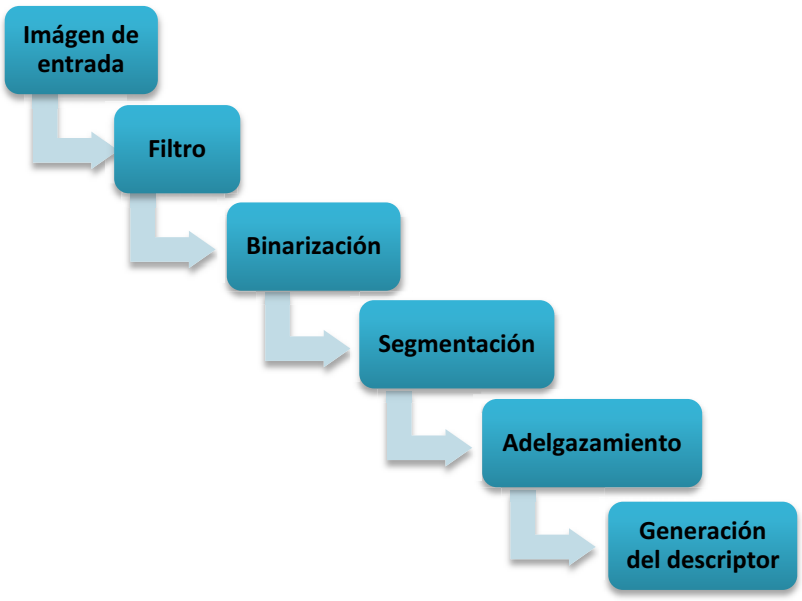


Diagrama 1. Procesos a los que se somete la imagen de entrada.

Módulo de filtrado de imagen

El objetivo es obtener una imagen con menos variabilidad en los valores de cada pixel, lo que ayudará a eliminar ruido en la imagen.

Proceso: se realiza un filtro gaussiano (también conocido como desenfoque gaussiano), el primer paso es calcular una máscara de filtro[6], después se recorre cada píxel de la matriz de la imagen y calcula el promedio de los valores RGB[7] de sus *n* vecinos próximos multiplicado por la máscara de filtro y este es el nuevo valor que se le asigna al píxel en proceso, un ejemplo de filtrado se muestra en la figura 2.

Figura 2. Proceso de filtrado gaussiano aplicado a la imagen de entrada de la izquierda y como resultado se obtiene la imagen de la derecha. Se puede describir como una difuminación de la imagen.

[6] Máscara de filtro: matriz de píxeles de tamaño n x m, el cual se elige de acuerdo al objetivo al que se deseé llegar.
[7] RGB: valor de intensidad para cada color en formato rojo, verde y azul (por sus iniciales en inglés), que describen a un píxel.

Para poder realizar este proceso es necesario aplicar una operación de convolución, dicha operación es calcular el valor de cada pixel de la nueva imagen obtenido mediante el promedio del producto de cada valor del pixel vecino por el valor correspondiente de su coordenada en la máscara de filtro.

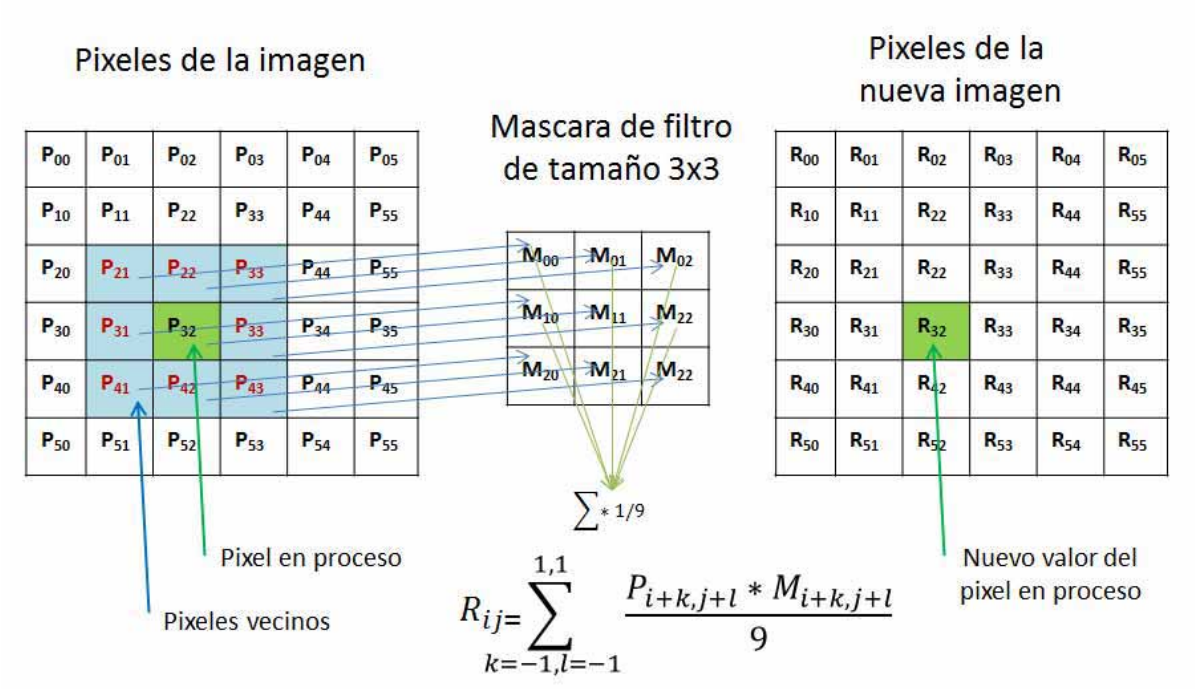


Figura 3. Se ejemplifica la operación de convolución con una máscara de filtro de tamaño 3x3.

Antes de seguir hay que mencionar que el radio es el tamaño de la máscara de filtro deseada, así como también la cota para definir el número de vecinos que se incluyen en la operación del pixel en proceso.

Para poder realizar el proceso primero se genera una máscara de filtro de acuerdo a los parámetros sigma (σ) y radio, dados por el usuario. Donde para cada valor de la matriz de la máscara de filtro se calcula con la ecuación de función gaussiana (Ec. 1)

$$M(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \text{Ec. 1.}$$

En donde:

σ es la desviación estándar de los valores de cada pixel en la imagen
x e y son las coordenadas de la celda a calcular.

Módulo de binarización de la imagen

Se tiene como objetivo convertir la representación de color de cada pixel de la imagen en negro o blanco (valores 0 o 255, respectivamente).

Proceso: se recorre píxel por píxel en la imagen y por medio de un umbral[8] se determina si el píxel en proceso se convierte a negro o blanco, dependiendo si el valor del píxel actual es mayor o menor a el umbral. La figura 4 muestra un ejemplo.

Figura 4. Ejemplo del proceso de binarización de una imagen.

Módulo de segmentación

El objetivo de este módulo es definir las regiones que componen a la imagen, haciendo agrupaciones de píxeles.

También llamado dilatación, lo que se hace es rellenar los huecos que hay en la imagen y agrandando regiones de pixeles que sean del mismo tipo. Ya que nuestro objetivo se enfoca en la forma que es visible en la imagen para la cual se desea obtener el contorno, entonces se desea que no tenga huecos la figura y que al pasar a la siguiente etapa este bien definida y delimitada.

Para poder hacer este proceso se realiza un barrido de cada pixel de la imagen y a través de un elemento estructurante (EE), se van rellenando o no los pixeles vecinos si corresponden al EE con el pixel en proceso.

En la figura 5 se puede observar cómo se hace un barrido con el EE cruz a la imagen original y los cuadros en color hueso son los que se mapean entre el pixel en proceso y el EE, entonces se dibujan los pixeles extras que abarquen al pixel en estudio en unión al EE, obteniendo como resultado la imagen nueva.

[8] Umbral: valor mínimo de una magnitud a partir del cual se produce un efecto determinado.

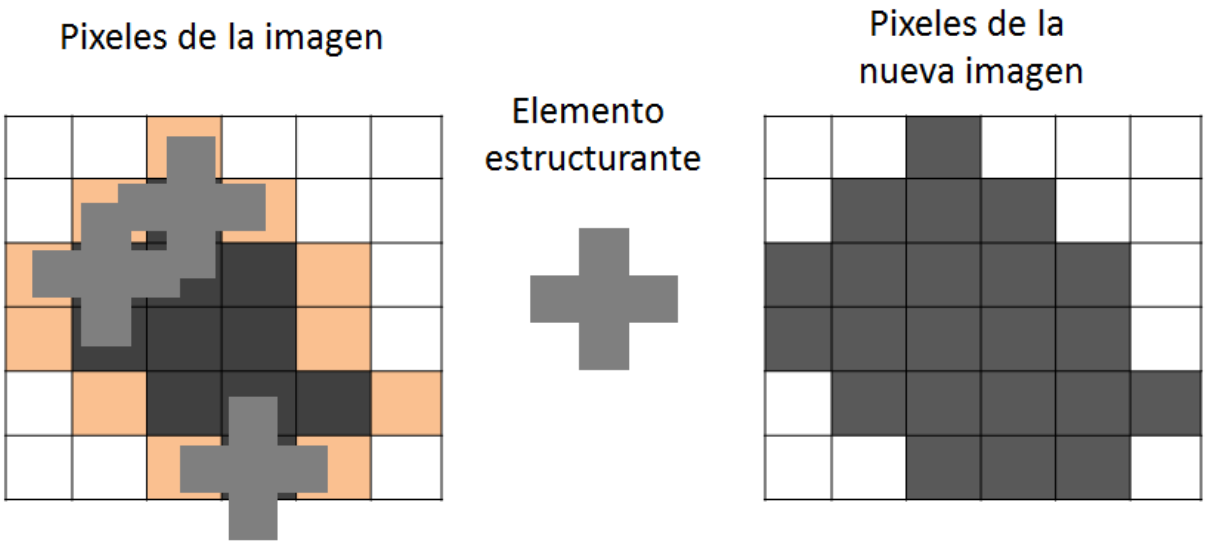


Figura 5. Operación de dilatación.

Hay que tomar en cuenta que el fondo de la imagen puede ser considerado a la hora de hacer la dilatación y esto generaría una solución errónea, por lo que se tiene que realizar un método que identifique el fondo de la imagen para que se excluya al momento de dilatarla. Este error podría ocasionar que no se identifique bien el contorno de la imagen y se tome en cuenta parte del fondo como parte de la figura en estudio.

Así entonces podremos obtener una imagen donde esté bien definida y delimitada la figura en estudio, y al no considerar el fondo se rellenarían los huecos solamente que se encuentren en el centro de la figura.

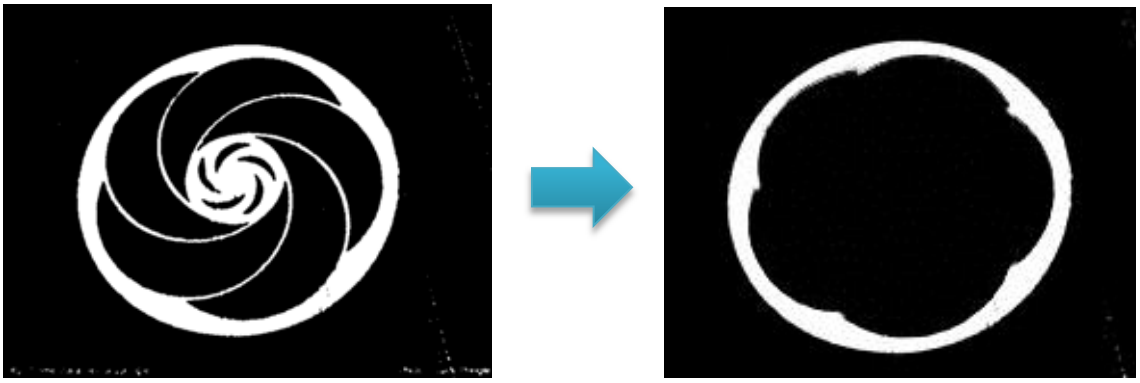


Figura 6. Ejemplo de segmentación de una imagen.

Módulo de adelgazamiento

El objetivo es obtener el borde que delimita el área resultante del proceso anterior.

Proceso: por medio de operadores básicos de álgebra de conjuntos[9] y topología[10] se realiza la resta píxel a píxel con un EE de píxeles dado, que permiten obtener el contorno de la forma resultante del proceso anterior con una longitud de 1 píxel de ancho. La figura 7 muestra un ejemplo de adelgazamiento.

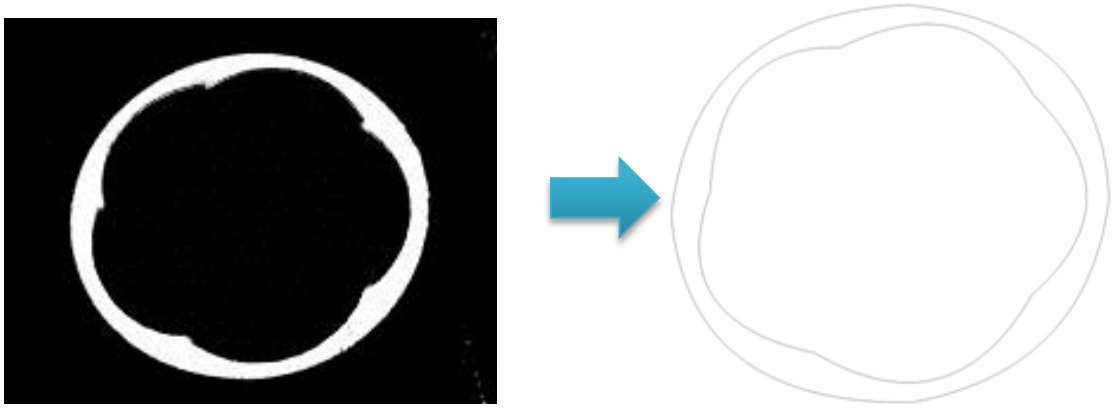


Figura 7. Ejemplo de adelgazamiento de una imagen.

Para poder llevar a cabo el proceso se tiene que haber obtenido una imagen perfectamente definida y delimitada del proceso anterior, ya que si no, entonces se estarían obteniendo objetos que no pertenecen a la figura.

La operación para generar el contorno de la imagen se deriva de la operación de dilatación, al igual se utilizó el mismo EE, y se cambian operaciones lógicas para que en vez de generar la imagen rellenada, solo se obtengan los pixeles nuevos a dibujar en la imagen después del barrido de cada pixel con el EE.

[9] Álgebra de conjuntos: operaciones básicas que se pueden realizar con conjuntos.
[10] Topología: rama de las matemáticas dedicada al estudio de propiedades de los cuerpos geométricos que no se alteran por transformaciones continuas.

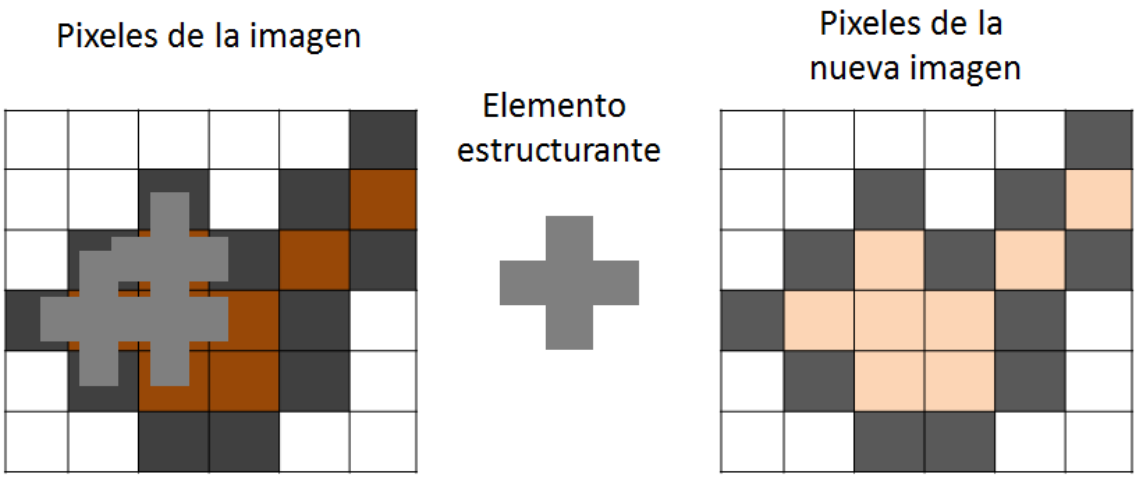


Figura 8. Operación de erosión para extraer el contorno.

En la figura 8 se puede observar cómo se hace un barrido con el EE cruz a la imagen original y los cuadros en color naranja son los que se mapean entre el pixel en proceso y el EE, entonces se dibujan de otro color los pixeles que mapeen totalmente con el EE, dejando así el contorno de ancho de un pixel del color original.

Módulo generador del descriptor

Se tiene como objetivo generar la matriz de píxeles que describan al contorno de la imagen. En donde se genera la imagen final de salida que muestra el contorno de la imagen inicial.

Conclusiones

En este proyecto se aplicaron métodos y técnicas de procesamiento digital que son muy conocidos en el ámbito del procesamiento digital de imágenes. En los cuales se aplican diferentes métodos y técnicas para obtener un resultado o modificación particular de la imagen, información que es de utilidad para la comunicación y buen funcionamiento ente etapas en secuencia, si en alguna de dichas etapas no se obtiene el mejor de los resultados para el objetivo a cumplir, se puede arrastrar el error hasta el final y obtener resultados no deseados.

Los procesos que se aplicaron en el proyecto se pueden considerar triviales pero tienen cierto grado de complejidad y es necesario conocer ciertos temas como mínimo de la materia. Un ejemplo de esto es el funcionamiento del filtro de Gauss, como bien se

sabe se tienen diversos usos de esta función, una distribución normal en probabilidad por ejemplo, y se puede observar como a raíz de un numero particular sus vecinos decaen rápidamente al alejarse de este número raíz, que a la vez se puede observar como el mayor.

Las técnicas aplicadas dan una idea global de lo que se estudia en dicha materia, así es de vital importancia que todo ingeniero en computación debería tener los conocimientos de que es lo que se estudia y desarrolla en el presente proyecto. Como ya se mencionó, son temas para los que un ingeniero en computación debería de ser básicos y conocidos, ya que son aplicados de diferente forma pero para situaciones similares.

Para el presente proyecto se pudieron aplicar métodos o técnicas de PDI más complicadas, o mejor dicho, más eficientes, pero esa sería tarea para algún otro alumno interesado en el tema que pueda mejorar alguna etapa del procesamiento de la imagen y en la cual se obtengan mejores resultado sin tanta variabilidad en los resultados.

Bibliografía

[1] B. James. (2013, Febrero 5). About the Crop Circles.[En línea].Disponible: http://www.temporarytemples.co.uk/faqs.

[2] M. E. García Avilés, "Software para validación antropométrica por procesamiento digital de imágenes", Proyecto Terminal, División de CBI, Universidad Autónoma Metropolitana Azcapotzalco, México, 2011.

[3] F. M. Díaz Cabrera. "Clasificador de objetos de banda infinita por medio de procesamiento digital de imágenes", Proyecto Terminal, División de CBI, Universidad Autónoma Metropolitana Azcapotzalco, México, 2009.

[4] J. R. Silva Guerrero. "Aproximación de la técnica de repixelización de arte pixelado", Propuesta de Proyecto Terminal, División de CBI, Universidad Autónoma Metropolitana Azcapotzalco, México, 2012.

[5] T. Sugiyama & K. Abe, "Edge feature analysis by a vectorized feature extractor and in multiple edges", IEEE Trans. Image Process, vol.2, no.,pp.280-284, Agosto 1996.

[6] Y. Wei, Z. Zhao, J. Song, "Urban building extraction from high-resolution satellite panchromatic image using clustering and edge detection", IEEE Trans. Image Process, vol.3, no.4,pp.2008-2010, Septiembre 2004.

[7] Q. Li, C. Kambhamettu, "Contour extraction of Drosophila embryos", IEEE/ACM Trans. Netw.,vol.8, no.6,pp.33-40, Noviembre 2011.

Apéndices

A continuación se tiene una breve descripción del código de la aplicación, y los módulos que la conforman, así como el código de la aplicación.

Diagrama de clases

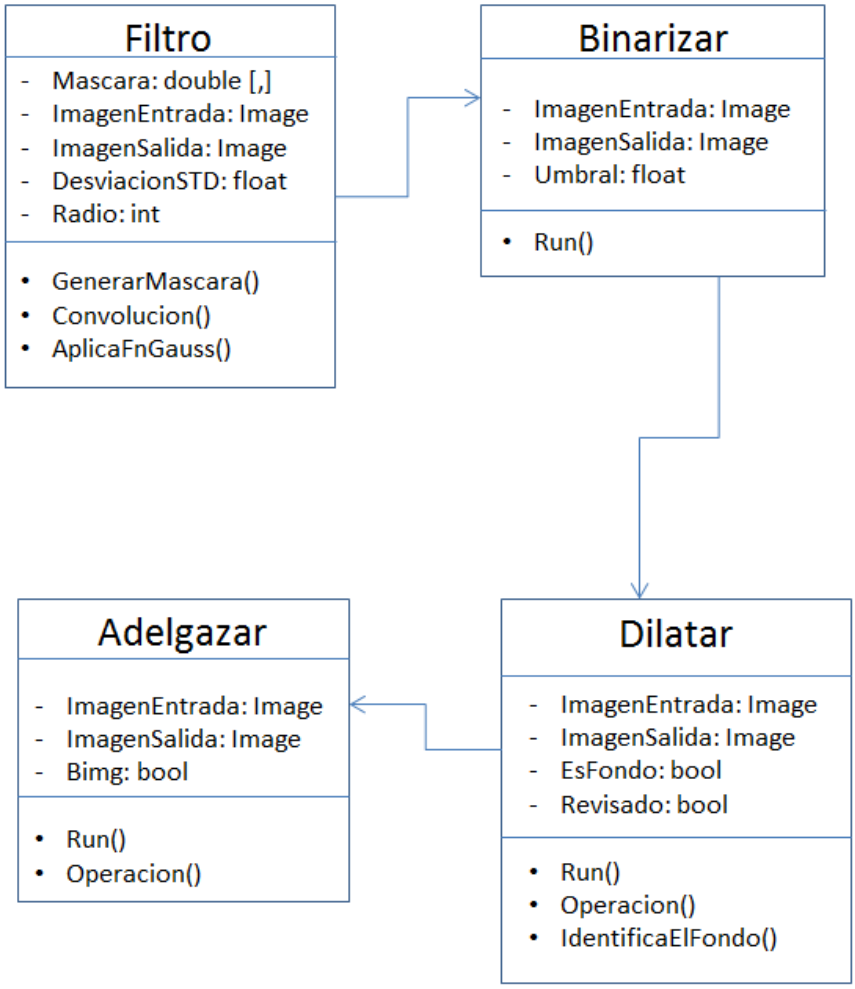
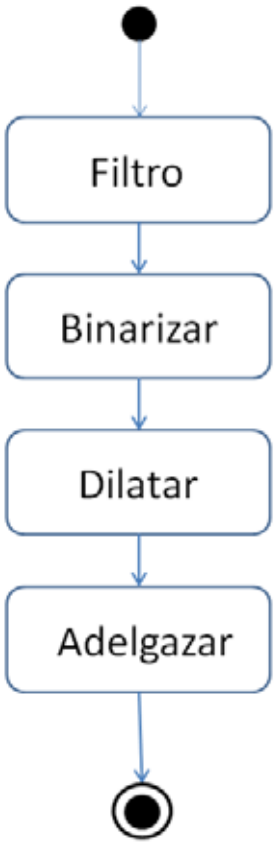


Diagrama de estados



Código de la aplicación

1. ***Clase Form1.***

   En esta clase se define la interfaz de usuario y la interacción con cada una de las clases donde se realizan las operaciones, así como la interfaz donde se muestra el resultado de cada una de las etapas de procesamiento. Además se controla la secuencia de los módulos y entradas y salidas de los parámetros.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Drawing.Imaging;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private Image ImgOrigFin, ImgFiltradaFin, ImgBinarizadaFin, ImgDilatadaFin,
ImgAdelgazadaFin, ImgBinarizadaFinTemp;
        private Boolean VacioOriginal = true, VacioFiltrada = true, VacioBinarizada
= true, VacioDilatada = true, VacioAdelgazada = true, Aviso = true;
        private void BTNCargarImg_Click(object sender, EventArgs e)
        {
                //Se delimitan los formatos aceptados para no generar error al
cargar un archivo no deseado
            openFileDialog1.Filter =
           "Images (*.BMP;*.JPG;*.PNG)|*.BMP;*.PNG;*.JPG|" +
           "All files (*.*)|*.*";
            openFileDialog1.Multiselect = true;
            openFileDialog1.Title = "Cargar imágen";
            openFileDialog1.ShowDialog();
        }

        //Función en la que se reduce la imágen si supera la especificación de 1000
x 1000 pixeles
        public static Image EscalarImagen(Image image, int maxWidth, int maxHeight)
        {
            var ratioX = (double)maxWidth / image.Width;
            var ratioY = (double)maxHeight / image.Height;
            var ratio = Math.Min(ratioX, ratioY);

            var newWidth = (int)(image.Width * ratio);
            var newHeight = (int)(image.Height * ratio);

            var newImage = new Bitmap(newWidth, newHeight);
            Graphics.FromImage(newImage).DrawImage(image, 0, 0, newWidth,
newHeight);
            return newImage;
        }

        private void flowLayoutPanel1_Paint(object sender, PaintEventArgs e)
```

```
        {        }

        private void tableLayoutPanel1_Paint(object sender, PaintEventArgs e)
        {        }

        private void Form1_Load(object sender, EventArgs e)
        {        }

        private void tabPage1_Click(object sender, EventArgs e)
        {        }

        private void button1_Click(object sender, EventArgs e)
        {
            tabControl1.SelectedIndex = 1;
        }
        //Función donde se controla la carga del archivo de imagen
        private void openFileDialog1_FileOk_1(object sender, CancelEventArgs e)
        {
            this.Activate();
            string[] files = openFileDialog1.FileNames;
            try
            {
                foreach (string file in files)
                {
                    System.IO.FileInfo fileInfo = new System.IO.FileInfo(file);
                    System.IO.FileStream fileStream = fileInfo.OpenRead();
                    ImgOrigFin = System.Drawing.Image.FromStream(fileStream);
                    Image ImgVisPrev;
                    if (ImgOrigFin.Height > 1000 || ImgOrigFin.Width > 1000)
                    {
                        if (MessageBox.Show("La imágen supera 1000 píxeles, ¿Desea
reducirla a una proporcion de 1000 pixeles?", "Advertencia",
MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.No)
                        {
                            break;
                        }
                        ImgOrigFin = EscalarImagen(ImgOrigFin, 1000, 1000);
                    }
                    ImgOrigFin.Save(@"c:\Original.png", ImageFormat.Png);
                    ImgVisPrev = ImgOrigFin;
                    String TextoImagen = "(Tamaño original)";
                    VacioOriginal = false;
                    if (ImgOrigFin.Height > 500 || ImgOrigFin.Width > 500)
                    {
                        ImgVisPrev = EscalarImagen(ImgOrigFin, 500, 500);
                        TextoImagen = "(Vista Previa)";
                    }
                    PICImagenIn.Image = ImgVisPrev;
                    PICImagenIn.Height = ImgVisPrev.Height;
                    PICImagenIn.Width = ImgVisPrev.Width;
                    label1.Text = TextoImagen + " Resolución: Ancho=" +
ImgVisPrev.Width + "p. Alto=" + ImgVisPrev.Height + "p.";
                    BtnSig.BackColor = System.Drawing.Color.GreenYellow;
                    this.tabPage2.Enabled = true;
                    //PICImagenIn.Enabled = false;
                    //reset a panel filtrado
                    VacioFiltrada = true;
                    button2.BackColor = System.Drawing.Color.Transparent;
```

```
                    PICImagenFiltrada.Image = null;
                    //reset a panel binarizado
                    this.tabPage3.Enabled = false;
                    button5.BackColor = System.Drawing.Color.Transparent;
                    PICImagenBinarizada.Image = null;
                    //reset a panel dilatado
                    this.tabPage4.Enabled = false;
                    button7.BackColor = System.Drawing.Color.Transparent;
                    PICImagenDilatada.Image = null;
                    //reset a panel adelgazamiento
                    this.tabPage5.Enabled = false;
                    BTNAdelgazar.Visible = true;
                    Finalizo.Visible = false;
                    PICImagenAdelgazada.Image = null;

                    Application.DoEvents();
                    fileStream.Close();
                }
            }
            catch (ArgumentException)
            {
                MessageBox.Show("Error: " +
                "Verifica el origen de la imagen");
            }


        }

        private void PICImagenIn_Click(object sender, EventArgs e)
        {
            if (!VacioOriginal)
            {
                Form2 ImgExpandida = new Form2();
                ImgExpandida.SetImgPicture(ImgOrigFin);
                ImgExpandida.Show();
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            if (Aviso)
            {
                if (MessageBox.Show("Verifica que la imágen se adecue a tu
propósito, para ello puedes manipular los valores Sigma y Radio.\n ¿Deseas pasar a
la siguiente etapa?", "Aviso", MessageBoxButtons.YesNo, MessageBoxIcon.Information)
== DialogResult.Yes)
                    tabControl1.SelectedIndex = 2;
                    Aviso = false;
            }
            else
            {
                tabControl1.SelectedIndex = 2;
                Aviso = true;
            }
        }

        private void button1_Click_1(object sender, EventArgs e)
        {
```

```
            tabControl1.SelectedIndex = 0;
        }

        private void BTNFiltro_Click(object sender, EventArgs e)
        {
            Image ImgVisPrev;
            float DesvStd = (float.Parse)(VALVarianza.Value.ToString());
            int Radio = (int.Parse)(VALRadio.Value.ToString());
            MascaraFiltro Mascara = new MascaraFiltro();
            if ((Radio % 2) == 0)   // de todos modos no aplica, se convierte cuando
se calcula la mitad de la dimesion de la mascara
                Radio++;
            ImgFiltradaFin = Mascara.Convolucion((Bitmap)ImgOrigFin, DesvStd,
Radio);
            ImgFiltradaFin.Save(@"c:\Filtrada.bmp", ImageFormat.Bmp);
            ImgVisPrev = EscalarImagen(ImgFiltradaFin, 500, 500);
            PICImagenFiltrada.Image = ImgVisPrev;
            PICImagenFiltrada.Height = ImgVisPrev.Height;
            PICImagenFiltrada.Width = ImgVisPrev.Width;
            button2.BackColor = System.Drawing.Color.GreenYellow;
            this.tabPage3.Enabled = true;
            VacioFiltrada = false;
        }

        private void button4_Click(object sender, EventArgs e)
        {
            tabControl1.SelectedIndex = 1;
        }



        private void button5_Click(object sender, EventArgs e)
        {
            if (MessageBox.Show("Verifica que la imágen se adecue a tu propósito,
para ello puedes manipular el valor de la variable Umbral.\n ¿Deseas pasar a la
siguiente etapa?", "Aviso", MessageBoxButtons.YesNo, MessageBoxIcon.Information) ==
DialogResult.Yes)
            {
                tabControl1.SelectedIndex = 3;
            }
        }

        private void BTNDilatar_Click_1(object sender, EventArgs e)
        {
            Image ImgVisPrev;
            Dilatar Dilatacion = new Dilatar();
            ImgDilatadaFin = Dilatacion.Run((Bitmap)ImgBinarizadaFinTemp);
            ImgBinarizadaFinTemp = ImgDilatadaFin;
            ImgDilatadaFin.Save(@"c:\Dilatada.bmp", ImageFormat.Bmp);
            ImgVisPrev = EscalarImagen(ImgDilatadaFin, 500, 500);
            PICImagenDilatada.Image = ImgVisPrev;
            PICImagenDilatada.Height = ImgVisPrev.Height;
            PICImagenDilatada.Width = ImgVisPrev.Width;
            button7.BackColor = System.Drawing.Color.GreenYellow;
            this.tabPage5.Enabled = true;
            VacioDilatada = false;
        }
```

```
        private void BTNBinarizar_Click(object sender, EventArgs e)
        {
            Image ImgVisPrev;
            float Umbral = (float.Parse)(ValUmbral.Value.ToString());
            Binarizar Binarizacion = new Binarizar();
            ImgBinarizadaFin = Binarizacion.Run((Bitmap)ImgFiltradaFin, Umbral);
            ImgBinarizadaFinTemp = ImgBinarizadaFin;
            ImgBinarizadaFin.Save(@"c:\Binarizada.bmp", ImageFormat.Bmp);
            ImgVisPrev = EscalarImagen(ImgBinarizadaFin, 500, 500);
            PICImagenBinarizada.Image = ImgVisPrev;
            PICImagenBinarizada.Height = ImgVisPrev.Height;
            PICImagenBinarizada.Width = ImgVisPrev.Width;
            button5.BackColor = System.Drawing.Color.GreenYellow;
            this.tabPage4.Enabled = true;
            VacioBinarizada = false;
        }

        private void BTNAdelgazar_Click(object sender, EventArgs e)
        {
            Image ImgVisPrev;
            Adelgazar Adelgazamiento = new Adelgazar();
            ImgAdelgazadaFin = Adelgazamiento.Run((Bitmap)ImgDilatadaFin);
            ImgAdelgazadaFin.Save(@"c:\Adelgazada.bmp", ImageFormat.Bmp);
            ImgVisPrev = EscalarImagen(ImgAdelgazadaFin, 500, 500);
            PICImagenAdelgazada.Image = ImgVisPrev;
            PICImagenAdelgazada.Height = ImgVisPrev.Height;
            PICImagenAdelgazada.Width = ImgVisPrev.Width;
            VacioAdelgazada = false;
            Finalizo.Visible = true;
            BTNAdelgazar.Visible = false;
            MessageBox.Show("El proceso ha finalizado exitosamente.", "Proceso
Finalizado");
        }


        private void button7_Click(object sender, EventArgs e)
        {
            if (MessageBox.Show("Verifica que la imágen se adecue a tu propósito,
para ello puedes volver a aplicar el "+
                "proceso a la imágen presionando el boton 'Inicar Proceso' .\n
¿Deseas pasar a la siguiente etapa?", "Aviso", MessageBoxButtons.YesNo,
MessageBoxIcon.Information) == DialogResult.Yes)
            {
                tabControl1.SelectedIndex = 4;
            }
        }

        private void PICImagenFiltrada_Click(object sender, EventArgs e)
        {
            if (!VacioFiltrada)
            {
                Form2 ImgExpandida = new Form2();
                ImgExpandida.SetImgPicture(ImgFiltradaFin);
                ImgExpandida.Show();
            }
        }

        private void PICImagenBinarizada_Click(object sender, EventArgs e)
```

```
        {
            if (!VacioBinarizada)
            {
                Form2 ImgExpandida = new Form2();
                ImgExpandida.SetImgPicture(ImgBinarizadaFin);
                ImgExpandida.Show();
            }

        }

        private void PICImagenDilatada_Click(object sender, EventArgs e)
        {
            if (!VacioDilatada)
            {
                Form2 ImgExpandida = new Form2();
                ImgExpandida.SetImgPicture(ImgDilatadaFin);
                ImgExpandida.Show();
            }
        }

        private void PICImagenAdelgazada_Click(object sender, EventArgs e)
        {
            if (!VacioAdelgazada)
            {
                Form2 ImgExpandida = new Form2();
                ImgExpandida.SetImgPicture(ImgAdelgazadaFin);
                ImgExpandida.Show();
            }
        }

    }
}
```

2. Clase Designer Form 1.

En esta clase se definen los componentes de la interfaz que serán de utilidad a lo largo de todo el proceso y etapas de la aplicación.

```
namespace WindowsFormsApplication1
{
    partial class Form1
    {
        /// <summary>
        /// Variable del diseñador requerida.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Limpiar los recursos que se estén utilizando.
        /// </summary>
        /// <param name="disposing">true si los recursos administrados se deben
eliminar; false en caso contrario.</param>
        protected override void Dispose(bool disposing)
```

```
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Código generado por el Diseñador de Windows Forms

        /// <summary>
        /// Método necesario para admitir el Diseñador. No se puede modificar
        /// el contenido del método con el editor de código.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.tabControl1 = new System.Windows.Forms.TabControl();
            this.tabPage1 = new System.Windows.Forms.TabPage();
            this.label1 = new System.Windows.Forms.Label();
            this.BTNCargarImg = new System.Windows.Forms.Button();
            this.BtnSig = new System.Windows.Forms.Button();
            this.PICImagenIn = new System.Windows.Forms.PictureBox();
            this.tabPage2 = new System.Windows.Forms.TabPage();
            this.BTNFiltro = new System.Windows.Forms.Button();
            this.label3 = new System.Windows.Forms.Label();
            this.VALRadio = new System.Windows.Forms.NumericUpDown();
            this.VALVarianza = new System.Windows.Forms.NumericUpDown();
            this.label2 = new System.Windows.Forms.Label();
            this.button1 = new System.Windows.Forms.Button();
            this.button2 = new System.Windows.Forms.Button();
            this.PICImagenFiltrada = new System.Windows.Forms.PictureBox();
            this.tabPage3 = new System.Windows.Forms.TabPage();
            this.BTNBinarizar = new System.Windows.Forms.Button();
            this.ValUmbral = new System.Windows.Forms.NumericUpDown();
            this.label4 = new System.Windows.Forms.Label();
            this.button4 = new System.Windows.Forms.Button();
            this.button5 = new System.Windows.Forms.Button();
            this.PICImagenBinarizada = new System.Windows.Forms.PictureBox();
            this.tabPage4 = new System.Windows.Forms.TabPage();
            this.BTNDilatar = new System.Windows.Forms.Button();
            this.button6 = new System.Windows.Forms.Button();
            this.button7 = new System.Windows.Forms.Button();
            this.PICImagenDilatada = new System.Windows.Forms.PictureBox();
            this.tabPage5 = new System.Windows.Forms.TabPage();
            this.Finalizo = new System.Windows.Forms.Label();
            this.PICImagenAdelgazada = new System.Windows.Forms.PictureBox();
            this.BTNAdelgazar = new System.Windows.Forms.Button();
            this.button8 = new System.Windows.Forms.Button();
            this.openFileDialog1 = new System.Windows.Forms.OpenFileDialog();
            this.toolTip1 = new System.Windows.Forms.ToolTip(this.components);
            this.tabControl1.SuspendLayout();
            this.tabPage1.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.PICImagenIn)).BeginInit();
            this.tabPage2.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.VALRadio)).BeginInit();
```

```
((System.ComponentModel.ISupportInitialize)(this.VALVarianza)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.PICImagenFiltrada)).BeginInit();
            this.tabPage3.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.ValUmbral)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.PICImagenBinarizada)).BeginInit();
            this.tabPage4.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.PICImagenDilatada)).BeginInit();
            this.tabPage5.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.PICImagenAdelgazada)).BeginInit();
            this.SuspendLayout();
            //
            // tabControl1
            //
            this.tabControl1.Controls.Add(this.tabPage1);
            this.tabControl1.Controls.Add(this.tabPage2);
            this.tabControl1.Controls.Add(this.tabPage3);
            this.tabControl1.Controls.Add(this.tabPage4);
            this.tabControl1.Controls.Add(this.tabPage5);
            this.tabControl1.Location = new System.Drawing.Point(26, 16);
            this.tabControl1.Name = "tabControl1";
            this.tabControl1.SelectedIndex = 0;
            this.tabControl1.Size = new System.Drawing.Size(580, 610);
            this.tabControl1.TabIndex = 1;
            //
            // tabPage1
            //
            this.tabPage1.Controls.Add(this.label1);
            this.tabPage1.Controls.Add(this.BTNCargarImg);
            this.tabPage1.Controls.Add(this.BtnSig);
            this.tabPage1.Controls.Add(this.PICImagenIn);
            this.tabPage1.Location = new System.Drawing.Point(4, 22);
            this.tabPage1.Name = "tabPage1";
            this.tabPage1.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage1.Size = new System.Drawing.Size(572, 584);
            this.tabPage1.TabIndex = 0;
            this.tabPage1.Text = "Carga de Imágen";
            this.tabPage1.UseVisualStyleBackColor = true;
            this.tabPage1.Click += new System.EventHandler(this.tabPage1_Click);
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(298, 562);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(0, 13);
            this.label1.TabIndex = 2;
            //
            // BTNCargarImg
            //
            this.BTNCargarImg.Location = new System.Drawing.Point(32, 24);
            this.BTNCargarImg.Name = "BTNCargarImg";
            this.BTNCargarImg.Size = new System.Drawing.Size(85, 23);
```

```
            this.BTNCargarImg.TabIndex = 3;
            this.BTNCargarImg.Tag = "";
            this.BTNCargarImg.Text = "Cargar Imágen";
            this.BTNCargarImg.UseVisualStyleBackColor = true;
            this.BTNCargarImg.Click += new
System.EventHandler(this.BTNCargarImg_Click);
            //
            // BtnSig
            //
            this.BtnSig.BackColor = System.Drawing.Color.Transparent;
            this.BtnSig.Location = new System.Drawing.Point(469, 24);
            this.BtnSig.Name = "BtnSig";
            this.BtnSig.Size = new System.Drawing.Size(63, 23);
            this.BtnSig.TabIndex = 1;
            this.BtnSig.Text = "Siguiente";
            this.BtnSig.UseVisualStyleBackColor = false;
            this.BtnSig.Click += new System.EventHandler(this.button1_Click);
            //
            // PICImagenIn
            //
            this.PICImagenIn.BorderStyle = System.Windows.Forms.BorderStyle.Fixed3D;
            this.PICImagenIn.Cursor = System.Windows.Forms.Cursors.Hand;
            this.PICImagenIn.Location = new System.Drawing.Point(32, 59);
            this.PICImagenIn.Name = "PICImagenIn";
            this.PICImagenIn.Size = new System.Drawing.Size(500, 500);
            this.PICImagenIn.TabIndex = 0;
            this.PICImagenIn.TabStop = false;
            this.PICImagenIn.Click += new
System.EventHandler(this.PICImagenIn_Click);
            //
            // tabPage2
            //
            this.tabPage2.Controls.Add(this.BTNFiltro);
            this.tabPage2.Controls.Add(this.label3);
            this.tabPage2.Controls.Add(this.VALRadio);
            this.tabPage2.Controls.Add(this.VALVarianza);
            this.tabPage2.Controls.Add(this.label2);
            this.tabPage2.Controls.Add(this.button1);
            this.tabPage2.Controls.Add(this.button2);
            this.tabPage2.Controls.Add(this.PICImagenFiltrada);
            this.tabPage2.Enabled = false;
            this.tabPage2.Location = new System.Drawing.Point(4, 22);
            this.tabPage2.Name = "tabPage2";
            this.tabPage2.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage2.Size = new System.Drawing.Size(572, 584);
            this.tabPage2.TabIndex = 1;
            this.tabPage2.Text = "Filtro de Gauss";
            this.tabPage2.UseVisualStyleBackColor = true;
            //
            // BTNFiltro
            //
            this.BTNFiltro.Location = new System.Drawing.Point(348, 14);
            this.BTNFiltro.Name = "BTNFiltro";
            this.BTNFiltro.Size = new System.Drawing.Size(67, 39);
            this.BTNFiltro.TabIndex = 11;
            this.BTNFiltro.Text = "Iniciar Proceso";
            this.BTNFiltro.UseVisualStyleBackColor = true;
            this.BTNFiltro.Click += new System.EventHandler(this.BTNFiltro_Click);
```

```
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Location = new System.Drawing.Point(235, 29);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(44, 13);
            this.label3.TabIndex = 10;
            this.label3.Text = "Radio =";
            //
            // VALRadio
            //
            this.VALRadio.InterceptArrowKeys = false;
            this.VALRadio.Location = new System.Drawing.Point(284, 26);
            this.VALRadio.Maximum = new decimal(new int[] {
            11,
            0,
            0,
            0});
            this.VALRadio.Minimum = new decimal(new int[] {
            3,
            0,
            0,
            0});
            this.VALRadio.Name = "VALRadio";
            this.VALRadio.Size = new System.Drawing.Size(32, 20);
            this.VALRadio.TabIndex = 9;
            this.VALRadio.TextAlign =
System.Windows.Forms.HorizontalAlignment.Center;
            this.VALRadio.Value = new decimal(new int[] {
            3,
            0,
            0,
            0});
            //
            // VALVarianza
            //
            this.VALVarianza.DecimalPlaces = 1;
            this.VALVarianza.Increment = new decimal(new int[] {
            5,
            0,
            0,
            65536});
            this.VALVarianza.Location = new System.Drawing.Point(169, 27);
            this.VALVarianza.Maximum = new decimal(new int[] {
            20,
            0,
            0,
            0});
            this.VALVarianza.Minimum = new decimal(new int[] {
            5,
            0,
            0,
            65536});
            this.VALVarianza.Name = "VALVarianza";
            this.VALVarianza.Size = new System.Drawing.Size(51, 20);
            this.VALVarianza.TabIndex = 8;
```

```
            this.VALVarianza.TextAlign =
System.Windows.Forms.HorizontalAlignment.Center;
            this.VALVarianza.Value = new decimal(new int[] {
            10,
            0,
            0,
            65536});
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(123, 29);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(45, 13);
            this.label2.TabIndex = 7;
            this.label2.Text = "Sigma =";
            //
            // button1
            //
            this.button1.Location = new System.Drawing.Point(32, 24);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(63, 23);
            this.button1.TabIndex = 6;
            this.button1.Text = "Anterior";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.button1_Click_1);
            //
            // button2
            //
            this.button2.Location = new System.Drawing.Point(469, 24);
            this.button2.Name = "button2";
            this.button2.Size = new System.Drawing.Size(63, 23);
            this.button2.TabIndex = 5;
            this.button2.Text = "Siguiente";
            this.button2.UseVisualStyleBackColor = true;
            this.button2.Click += new System.EventHandler(this.button2_Click);
            //
            // PICImagenFiltrada
            //
            this.PICImagenFiltrada.BorderStyle =
System.Windows.Forms.BorderStyle.Fixed3D;
            this.PICImagenFiltrada.Cursor = System.Windows.Forms.Cursors.Hand;
            this.PICImagenFiltrada.Location = new System.Drawing.Point(32, 59);
            this.PICImagenFiltrada.Name = "PICImagenFiltrada";
            this.PICImagenFiltrada.Size = new System.Drawing.Size(500, 500);
            this.PICImagenFiltrada.TabIndex = 4;
            this.PICImagenFiltrada.TabStop = false;
            this.PICImagenFiltrada.Click += new
System.EventHandler(this.PICImagenFiltrada_Click);
            //
            // tabPage3
            //
            this.tabPage3.Controls.Add(this.BTNBinarizar);
            this.tabPage3.Controls.Add(this.ValUmbral);
            this.tabPage3.Controls.Add(this.label4);
            this.tabPage3.Controls.Add(this.button4);
            this.tabPage3.Controls.Add(this.button5);
            this.tabPage3.Controls.Add(this.PICImagenBinarizada);
```

```
            this.tabPage3.Enabled = false;
            this.tabPage3.Location = new System.Drawing.Point(4, 22);
            this.tabPage3.Name = "tabPage3";
            this.tabPage3.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage3.Size = new System.Drawing.Size(572, 584);
            this.tabPage3.TabIndex = 2;
            this.tabPage3.Text = "Binarización";
            this.tabPage3.UseVisualStyleBackColor = true;
            //
            // BTNBinarizar
            //
            this.BTNBinarizar.Location = new System.Drawing.Point(348, 14);
            this.BTNBinarizar.Name = "BTNBinarizar";
            this.BTNBinarizar.Size = new System.Drawing.Size(67, 39);
            this.BTNBinarizar.TabIndex = 18;
            this.BTNBinarizar.Text = "Iniciar Proceso";
            this.BTNBinarizar.UseVisualStyleBackColor = true;
            this.BTNBinarizar.Click += new
System.EventHandler(this.BTNBinarizar_Click);
            //
            // ValUmbral
            //
            this.ValUmbral.Location = new System.Drawing.Point(217, 27);
            this.ValUmbral.Maximum = new decimal(new int[] {
            250,
            0,
            0,
            0});
            this.ValUmbral.Minimum = new decimal(new int[] {
            50,
            0,
            0,
            0});
            this.ValUmbral.Name = "ValUmbral";
            this.ValUmbral.Size = new System.Drawing.Size(43, 20);
            this.ValUmbral.TabIndex = 17;
            this.ValUmbral.TextAlign =
System.Windows.Forms.HorizontalAlignment.Center;
            this.ValUmbral.Value = new decimal(new int[] {
            50,
            0,
            0,
            0});
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Location = new System.Drawing.Point(151, 29);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(49, 13);
            this.label4.TabIndex = 16;
            this.label4.Text = "Umbral =";
            //
            // button4
            //
            this.button4.Location = new System.Drawing.Point(32, 24);
            this.button4.Name = "button4";
            this.button4.Size = new System.Drawing.Size(63, 23);
```

```
            this.button4.TabIndex = 14;
            this.button4.Text = "Anterior";
            this.button4.UseVisualStyleBackColor = true;
            this.button4.Click += new System.EventHandler(this.button4_Click);
            //
            // button5
            //
            this.button5.Location = new System.Drawing.Point(469, 24);
            this.button5.Name = "button5";
            this.button5.Size = new System.Drawing.Size(63, 23);
            this.button5.TabIndex = 13;
            this.button5.Text = "Siguiente";
            this.button5.UseVisualStyleBackColor = true;
            this.button5.Click += new System.EventHandler(this.button5_Click);
            //
            // PICImagenBinarizada
            //
            this.PICImagenBinarizada.BorderStyle =
System.Windows.Forms.BorderStyle.Fixed3D;
            this.PICImagenBinarizada.Cursor = System.Windows.Forms.Cursors.Hand;
            this.PICImagenBinarizada.Location = new System.Drawing.Point(32, 59);
            this.PICImagenBinarizada.Name = "PICImagenBinarizada";
            this.PICImagenBinarizada.Size = new System.Drawing.Size(500, 500);
            this.PICImagenBinarizada.TabIndex = 12;
            this.PICImagenBinarizada.TabStop = false;
            this.PICImagenBinarizada.Click += new
System.EventHandler(this.PICImagenBinarizada_Click);
            //
            // tabPage4
            //
            this.tabPage4.Controls.Add(this.BTNDilatar);
            this.tabPage4.Controls.Add(this.button6);
            this.tabPage4.Controls.Add(this.button7);
            this.tabPage4.Controls.Add(this.PICImagenDilatada);
            this.tabPage4.Enabled = false;
            this.tabPage4.Location = new System.Drawing.Point(4, 22);
            this.tabPage4.Name = "tabPage4";
            this.tabPage4.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage4.Size = new System.Drawing.Size(572, 584);
            this.tabPage4.TabIndex = 3;
            this.tabPage4.Text = "Segmentación";
            this.tabPage4.UseVisualStyleBackColor = true;
            //
            // BTNDilatar
            //
            this.BTNDilatar.Location = new System.Drawing.Point(348, 14);
            this.BTNDilatar.Name = "BTNDilatar";
            this.BTNDilatar.Size = new System.Drawing.Size(67, 39);
            this.BTNDilatar.TabIndex = 19;
            this.BTNDilatar.Text = "Iniciar Proceso";
            this.BTNDilatar.UseVisualStyleBackColor = true;
            this.BTNDilatar.Click += new
System.EventHandler(this.BTNDilatar_Click_1);
            //
            // button6
            //
            this.button6.Location = new System.Drawing.Point(32, 24);
            this.button6.Name = "button6";
```

```
            this.button6.Size = new System.Drawing.Size(63, 23);
            this.button6.TabIndex = 17;
            this.button6.Text = "Anterior";
            this.button6.UseVisualStyleBackColor = true;
            //
            // button7
            //
            this.button7.Location = new System.Drawing.Point(469, 24);
            this.button7.Name = "button7";
            this.button7.Size = new System.Drawing.Size(63, 23);
            this.button7.TabIndex = 16;
            this.button7.Text = "Siguiente";
            this.button7.UseVisualStyleBackColor = true;
            this.button7.Click += new System.EventHandler(this.button7_Click);
            //
            // PICImagenDilatada
            //
            this.PICImagenDilatada.BorderStyle =
System.Windows.Forms.BorderStyle.Fixed3D;
            this.PICImagenDilatada.Cursor = System.Windows.Forms.Cursors.Hand;
            this.PICImagenDilatada.Location = new System.Drawing.Point(32, 59);
            this.PICImagenDilatada.Name = "PICImagenDilatada";
            this.PICImagenDilatada.Size = new System.Drawing.Size(500, 500);
            this.PICImagenDilatada.TabIndex = 15;
            this.PICImagenDilatada.TabStop = false;
            this.PICImagenDilatada.Click += new
System.EventHandler(this.PICImagenDilatada_Click);
            //
            // tabPage5
            //
            this.tabPage5.Controls.Add(this.Finalizo);
            this.tabPage5.Controls.Add(this.PICImagenAdelgazada);
            this.tabPage5.Controls.Add(this.BTNAdelgazar);
            this.tabPage5.Controls.Add(this.button8);
            this.tabPage5.Enabled = false;
            this.tabPage5.Location = new System.Drawing.Point(4, 22);
            this.tabPage5.Name = "tabPage5";
            this.tabPage5.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage5.Size = new System.Drawing.Size(572, 584);
            this.tabPage5.TabIndex = 4;
            this.tabPage5.Text = "Adelgazamiento";
            this.tabPage5.UseVisualStyleBackColor = true;
            //
            // Finalizo
            //
            this.Finalizo.AutoSize = true;
            this.Finalizo.BackColor = System.Drawing.Color.LawnGreen;
            this.Finalizo.Font = new System.Drawing.Font("Microsoft Sans Serif",
16.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.Finalizo.Location = new System.Drawing.Point(241, 19);
            this.Finalizo.Name = "Finalizo";
            this.Finalizo.Size = new System.Drawing.Size(260, 26);
            this.Finalizo.TabIndex = 23;
            this.Finalizo.Text = "¡El proceso ha terminado!";
            this.Finalizo.Visible = false;
            //
            // PICImagenAdelgazada
```

```
            //
            this.PICImagenAdelgazada.BorderStyle =
System.Windows.Forms.BorderStyle.Fixed3D;
            this.PICImagenAdelgazada.Cursor = System.Windows.Forms.Cursors.Hand;
            this.PICImagenAdelgazada.Location = new System.Drawing.Point(32, 59);
            this.PICImagenAdelgazada.Name = "PICImagenAdelgazada";
            this.PICImagenAdelgazada.Size = new System.Drawing.Size(500, 500);
            this.PICImagenAdelgazada.TabIndex = 22;
            this.PICImagenAdelgazada.TabStop = false;
            this.PICImagenAdelgazada.Click += new
System.EventHandler(this.PICImagenAdelgazada_Click);
            //
            // BTNAdelgazar
            //
            this.BTNAdelgazar.Location = new System.Drawing.Point(348, 14);
            this.BTNAdelgazar.Name = "BTNAdelgazar";
            this.BTNAdelgazar.Size = new System.Drawing.Size(67, 39);
            this.BTNAdelgazar.TabIndex = 21;
            this.BTNAdelgazar.Text = "Iniciar Proceso";
            this.BTNAdelgazar.UseVisualStyleBackColor = true;
            this.BTNAdelgazar.Click += new
System.EventHandler(this.BTNAdelgazar_Click);
            //
            // button8
            //
            this.button8.Location = new System.Drawing.Point(32, 24);
            this.button8.Name = "button8";
            this.button8.Size = new System.Drawing.Size(63, 23);
            this.button8.TabIndex = 20;
            this.button8.Text = "Anterior";
            this.button8.UseVisualStyleBackColor = true;
            //
            // openFileDialog1
            //
            this.openFileDialog1.FileName = "openFileDialog1";
            this.openFileDialog1.FileOk += new
System.ComponentModel.CancelEventHandler(this.openFileDialog1_FileOk_1);
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(634, 642);
            this.Controls.Add(this.tabControl1);
            this.Name = "Form1";
            this.Text = "Form1";
            this.Load += new System.EventHandler(this.Form1_Load);
            this.tabControl1.ResumeLayout(false);
            this.tabPage1.ResumeLayout(false);
            this.tabPage1.PerformLayout();

((System.ComponentModel.ISupportInitialize)(this.PICImagenIn)).EndInit();
            this.tabPage2.ResumeLayout(false);
            this.tabPage2.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.VALRadio)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.VALVarianza)).EndInit();
```

```
((System.ComponentModel.ISupportInitialize)(this.PICImagenFiltrada)).EndInit();
            this.tabPage3.ResumeLayout(false);
            this.tabPage3.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.ValUmbral)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.PICImagenBinarizada)).EndInit();
            this.tabPage4.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.PICImagenDilatada)).EndInit();
            this.tabPage5.ResumeLayout(false);
            this.tabPage5.PerformLayout();

((System.ComponentModel.ISupportInitialize)(this.PICImagenAdelgazada)).EndInit();
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.TabControl tabControl1;
        private System.Windows.Forms.TabPage tabPage1;
        private System.Windows.Forms.TabPage tabPage2;
        private System.Windows.Forms.PictureBox PICImagenIn;
        private System.Windows.Forms.TabPage tabPage3;
        private System.Windows.Forms.Button BtnSig;
        private System.Windows.Forms.Button BTNCargarImg;
        private System.Windows.Forms.OpenFileDialog openFileDialog1;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.PictureBox PICImagenFiltrada;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.NumericUpDown VALRadio;
        private System.Windows.Forms.NumericUpDown VALVarianza;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Button BTNFiltro;
        private System.Windows.Forms.Button button4;
        private System.Windows.Forms.Button button5;
        private System.Windows.Forms.PictureBox PICImagenBinarizada;
        private System.Windows.Forms.NumericUpDown ValUmbral;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.TabPage tabPage4;
        private System.Windows.Forms.TabPage tabPage5;
        private System.Windows.Forms.Button button6;
        private System.Windows.Forms.Button button7;
        private System.Windows.Forms.PictureBox PICImagenDilatada;
        private System.Windows.Forms.Button button8;
        private System.Windows.Forms.Button BTNDilatar;
        private System.Windows.Forms.Button BTNBinarizar;
        private System.Windows.Forms.ToolTip toolTip1;
        private System.Windows.Forms.Button BTNAdelgazar;
        private System.Windows.Forms.PictureBox PICImagenAdelgazada;
        private System.Windows.Forms.Label Finalizo;

    }
}
```

3. *Clase para realizar Filtro de Gauss.*

En esta clase se realizan las operaciones correspondientes al filtrado de gauss que se aplican a la imagen de entrada original y se envían al modulo de binarizado para su procesamiento posterior.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;

namespace WindowsFormsApplication1
{
    class MascaraFiltro
    {
        private double[,] Mascara;
        //En esta funcion se genera la mascara de filtro de tamaño radio x radio
        private void GenerarMascara(float DesviacionStd,int radio)
        {
            Mascara = new double[radio, radio];
            int x=0, y=0;
            Console.WriteLine("EL RADIO ES =" + radio);
            for (int i = -(int)(radio / 2); i <= (int)(radio / 2); i++)
            {
                for (int j = -(int)(radio / 2); j <= (int)(radio / 2); j++)
                {
                    Mascara[x, y] = AplicaFnGauss(i, j, DesviacionStd);
                    Console.WriteLine("Mascara[" + x + "][" + y + "]" + "=" +
Mascara[x, y]);
                    y++;
                }
                x++;
                y = 0;
            }
            int Radical = (int)(1 / Mascara[0, 0]);
            Console.WriteLine("el radical es="+Radical);
            for (int k = 0; k < radio; k++)
                for (int kk = 0; kk < radio; kk++)
                {
                    Mascara[k, kk] = Mascara[k, kk]*Radical;
                    Console.WriteLine("*** MascaraNormalizada[" + k + "][" + kk +
"]" + "=" + Mascara[k, kk]);
                }
        }
        //Se realiza la opracion de convolución pixel por pixel de la imagen
        public Bitmap Convolucion(Bitmap ImagenEntrada, float DesviacionSTD,int
radio)
        {
            int AnchoIMG = ImagenEntrada.Width;
            int AltoIMG = ImagenEntrada.Height;
            int radioTEMP =radio;
            Bitmap ImagenSalida = new Bitmap(AnchoIMG, AltoIMG);
```

```
            //Se manda a llamar a la función de generación de máscara con el radio y
desviación elegido
            GenerarMascara(DesviacionSTD,radio);
            int x, y;
            for (x = 0; x < AnchoIMG; x++)
            {
                for (y = 0; y < AltoIMG; y++)
                {
                    float PromR = 0, PromG = 0, PromB = 0;
                    Color pixelColor = ImagenEntrada.GetPixel(x, y);
                    int ii = x - (int)(radio / 2);
                    int jj = y - (int)(radio / 2);
                    for (int i = 0; i <= radio - 1; i++)
                    {
                        for (int j = 0; j <= radio - 1; j++)
                        {
                            Color ColorTemp;
                            if (ii < 0 || ii >= AnchoIMG || jj < 0 || jj >= AltoIMG)
                            {
                                ColorTemp = Color.FromArgb(0, 0, 0);
                            }
                            else
                            {
                                ColorTemp = ImagenEntrada.GetPixel(ii, jj);
                            }
                                PromR += (float)((Mascara[i, j]) * ColorTemp.R) /
(radio * radio);
                                PromG += (float)((Mascara[i, j]) * ColorTemp.G) /
(radio * radio);
                                PromB += (float)((Mascara[i, j]) * ColorTemp.B) /
(radio * radio);
                            jj++;
                        }
                        ii++;
                        jj = y - (int)(radio / 2);
                    }
                    //Se delimitan lo valores del promedio para no superar los
permitidos en el pixel (0-255)
                    if (PromR > 255 || PromR < 0)
                    {
                        if (PromR > 255)
                            PromR = 255;
                        else PromR = 0;
                    }
                    if (PromG > 255 || PromG < 0)
                    {
                        if (PromG > 255)
                            PromG = 255;
                        else PromG = 0;
                    }
                    if (PromB > 255 || PromB < 0)
                    {
                        if (PromB > 255)
                            PromB = 255;
                        else PromB = 0;
                    }
                    Color newColor = Color.FromArgb((int)PromR, (int)PromG,
(int)PromB);
```

```
                    ImagenSalida.SetPixel(x, y, newColor);
                }
            }


            return ImagenSalida;
        }
        //Función en la que se calcula el valor correspondiente a cada celda en la
máscara de guss
        private double AplicaFnGauss(int x, int y, float DesviacionStd)
        {
            double Gxy = 0, dividendo = 0, divisor = 0;
            dividendo = (Math.Exp(-((Math.Pow(x, 2.0) + Math.Pow(y, 2.0)) / (2 *
Math.Pow(DesviacionStd, 2.0)))));
            divisor = 1/(2 * Math.PI * Math.Pow(DesviacionStd, 2.0));
            Gxy = dividendo * divisor;
            return Gxy;
        }
    }
}
```

4. *Clase para binarizar la imagen*

En esta clase se realizan las operaciones que corresponden a la etapa de binarizado en las cuales se tiene como entrada la imagen procesada por la etapa de filtro de gauss y como salida la imagen binarizada que se envía a el modulo de dilatación.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;

namespace WindowsFormsApplication1
{
    class Binarizar
    {
        public Bitmap Run(Bitmap ImagenEntrada, float Umbral)
        {
            Bitmap ImagenSalida = new Bitmap(ImagenEntrada.Width,
ImagenEntrada.Height);
            for (int x = 0; x < ImagenEntrada.Width; x++)
            {
                for (int y = 0; y < ImagenEntrada.Height; y++)
                {
                    Color pixelColor = ImagenEntrada.GetPixel(x, y);
                    Color newColor;
                    //Se estudia a cada pixel y si es menor al umbral se le asigna
el valor de cero
                    if (pixelColor.R < Umbral & pixelColor.G < Umbral & pixelColor.B
< Umbral)
```

```
                    {
                        newColor = Color.FromArgb(0, 0, 0);

                    }
                    else
                    {
                        newColor = Color.FromArgb(255, 255, 255);
                    }
                    ImagenSalida.SetPixel(x, y, newColor);
                }
            }
            return ImagenSalida;
        }
    }
}
```

5. *Clase para dilatar la imagen*

Para esta clase se tienen las operaciones que permiten que la imagen se dilate, además se realiza la identificación del fondo para no considerarlo en la dilatación y que no se haga un cierre o se traslape el fondo con la figura de interés.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Collections;
using System.Drawing.Imaging;
using System.Threading.Tasks;
using System.Drawing;

namespace WindowsFormsApplication1
{
    class Dilatar
    {
        private Bitmap ImagenSalida, ImagenEntradaTemp;
        private bool[,] EsFondo;
        private bool[,] Revisado;

        public Bitmap Run(Bitmap ImagenEntrada)
        {
            EsFondo = new bool[ImagenEntrada.Width, ImagenEntrada.Height];
            Revisado = new bool[ImagenEntrada.Width, ImagenEntrada.Height];
            int i, j;
            for (i = 0; i < 4; i++)
                for (j = 0; j < 4; j++)
                {
                    EsFondo[i, j] = true;
                    Revisado[i, j] = true;
                }

            ImagenEntradaTemp = ImagenEntrada;
```

```
            ImagenSalida = new Bitmap(ImagenEntrada.Width, ImagenEntrada.Height);

            for (int x = 0; x < ImagenEntrada.Width; x++)
                for (int y = 0; y < ImagenEntrada.Height; y++)
                {
                    Color newColor = Color.FromArgb(255, 255, 255);
                    ImagenSalida.SetPixel(x, y, newColor);
                }
            //Se manda a llamar a la función de identificación de fondo para cada
pixel
            IdentificaElFondo();
            for (int x = 0; x < ImagenEntrada.Width; x++)
            {
                for (int y = 0; y < ImagenEntrada.Height; y++)
                {
                    // *
                    //***
                    // *
                    // ciclo que trabaja con el Elemento Estructurante anterior EE
cruz
                    Color pixelColor = ImagenEntradaTemp.GetPixel(x, y);
                    if (pixelColor.G == 0 && !EsFondo[x, y])
                    {
                        Operacion(x, y - 1);
                        Operacion(x - 1, y);
                        Operacion(x + 1, y);
                        Operacion(x, y + 1);
                    }
                }
            }
            return ImagenSalida;
        }
        //Se realiza el mapeo de cada pixel en la imagen con el EE cruz
        public void Operacion(int x, int y)
        {
            bool Bimg = false;

            if (x >= 0 && y >= 0)
                if (x < ImagenEntradaTemp.Width && y < ImagenEntradaTemp.Height)
                {
                    Color pixelColor = ImagenEntradaTemp.GetPixel(x, y);
                    if (pixelColor.R == 0)
                        Bimg = true;
                    else { }
                    bool Bres = false;
                    Bres = Bimg || true;
                    Color newColor;
                    if (Bres)
                    {
                        newColor = Color.FromArgb(0, 0, 0);
                    }
                    else
                    {
                        newColor = Color.FromArgb(255, 255, 255);
                    }

                    ImagenSalida.SetPixel(x, y, newColor);
                }
```

```
        }
        //Esta función identifica que pixeles de la imagen corresponden al fondo y
cuales no
        private void IdentificaElFondo()
        {
            Color newColor;
            int x, y;
            for (x = 1; x < ImagenEntradaTemp.Width - 1; x++)
                for (y = 1; y < ImagenEntradaTemp.Height - 1; y++)
                {
                    Color pixelColor = ImagenEntradaTemp.GetPixel(x, y);
                    if (pixelColor.G == 0)
                       if (EsFondo[x, y - 1] || EsFondo[x - 1, y] || EsFondo[x + 1,
y] || EsFondo[x, y + 1])
                        {
                            EsFondo[x, y] = true;
                            newColor = Color.FromArgb(0, 0, 0);
                            ImagenSalida.SetPixel(x, y, newColor);
                        }
                }

        }
    }
}
```

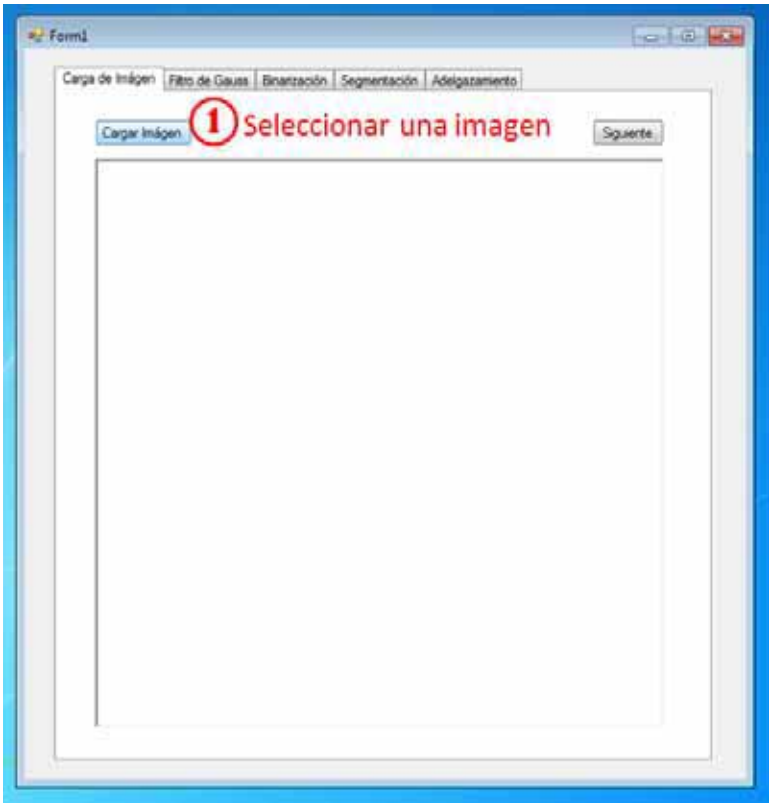
6. *Clase para dilatar la imagen.*

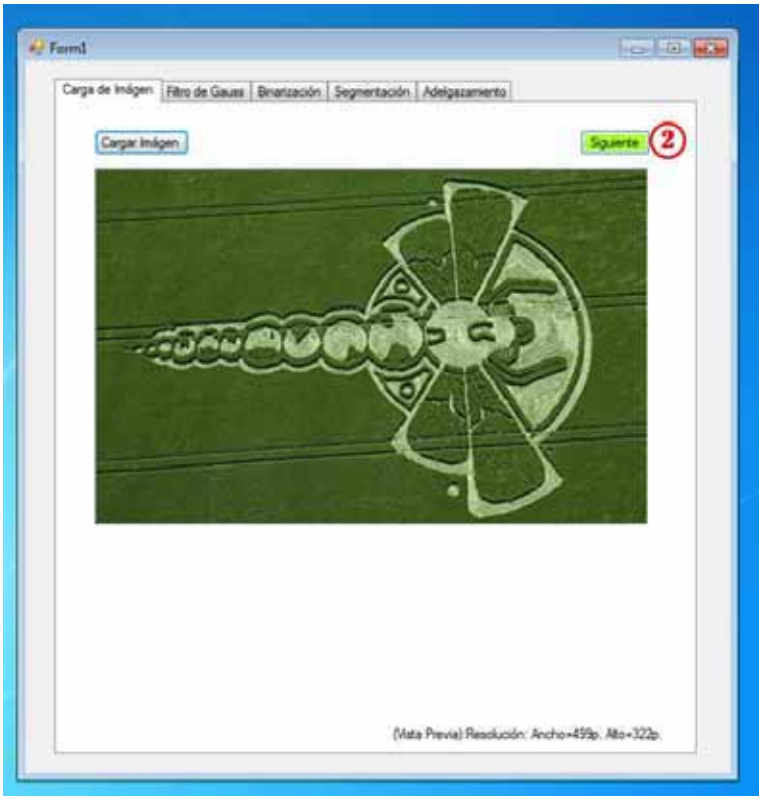
Es la clase donde se realizan las operaciones para obtener el contorno de la imagen.

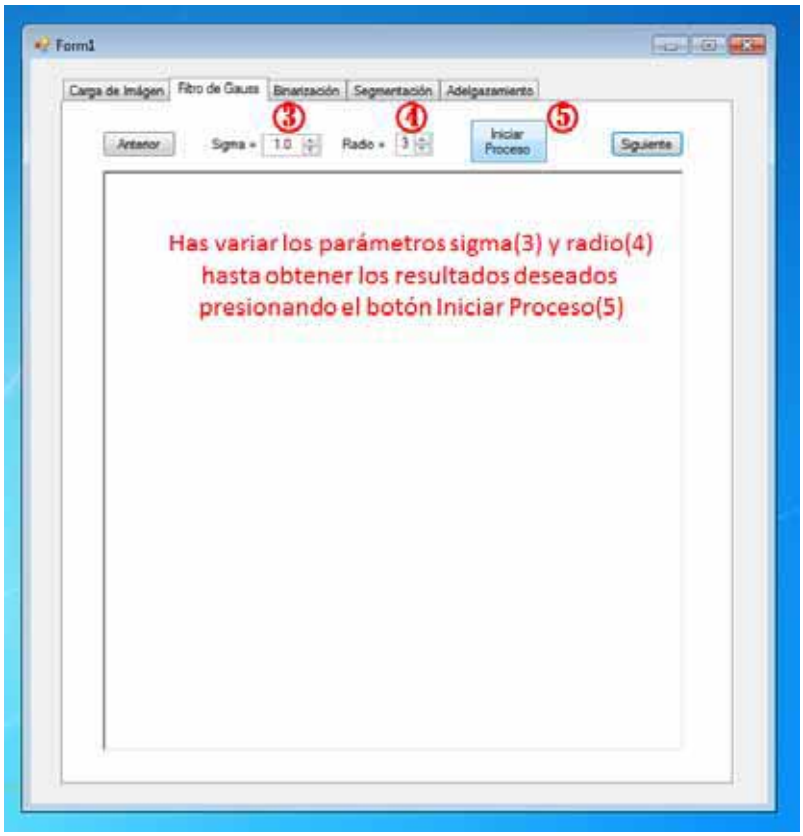
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;

namespace WindowsFormsApplication1
{
    class Adelgazar
    {

        private Bitmap ImagenSalida, ImagenEntradaTemp;
        //Se inicializan los valores de las variables temporales
        public Bitmap Run(Bitmap ImagenEntrada)
        {
            ImagenEntradaTemp = ImagenEntrada;
            ImagenSalida = new Bitmap(ImagenEntrada.Width, ImagenEntrada.Height);
            for (int x = 0; x < ImagenEntrada.Width; x++)
                for (int y = 0; y < ImagenEntrada.Height; y++)
                {
                    Color newColor = Color.FromArgb(255, 255, 255);
                    ImagenSalida.SetPixel(x, y, newColor);
                }
```

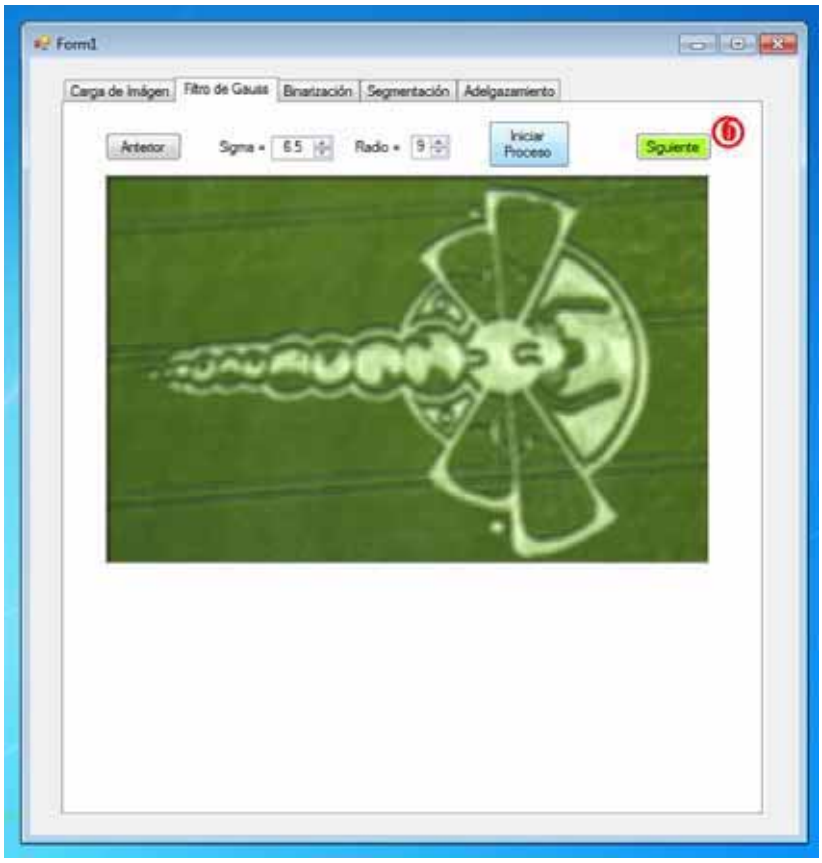
```
            for (int x = 0; x < ImagenEntrada.Width; x++)
            {
                for (int y = 0; y < ImagenEntrada.Height; y++)
                {
                    //***
                    //***
                    //***
                    // ciclo que trabaja con el Elemento Estructurante anterior
                    Color pixelColor = ImagenEntradaTemp.GetPixel(x, y);
                    if (pixelColor.G == 0)
                    {
                        for (int i = -1; i <= 1; i++)
                            for (int j = -1; j <= 1; j++)
                               Operacion(x+i, y+j);
                    }
                }
            }
            return ImagenSalida;
        }
        //Se realiza la operación pixel por pixel mapeando el EE con cada pixel en
la imagen
        public void Operacion(int x, int y)
        {
            bool Bimg = false;

            if (x >= 0 && y >= 0)
                if (x < ImagenEntradaTemp.Width && y < ImagenEntradaTemp.Height)
                {
                    Color pixelColor = ImagenEntradaTemp.GetPixel(x, y);
                    if (pixelColor.R == 0)
                        Bimg = true;
                    else { }
                    bool Bres = false;

                    Bres = Bimg && true;
                    Color newColor;
                    if (Bres)
                    {
                        newColor = Color.FromArgb(255, 255, 255);
                    }
                    else
                    {
                        newColor = Color.FromArgb(0, 0, 0);
                    }

                    ImagenSalida.SetPixel(x, y, newColor);
                }
        }

    }
}
```

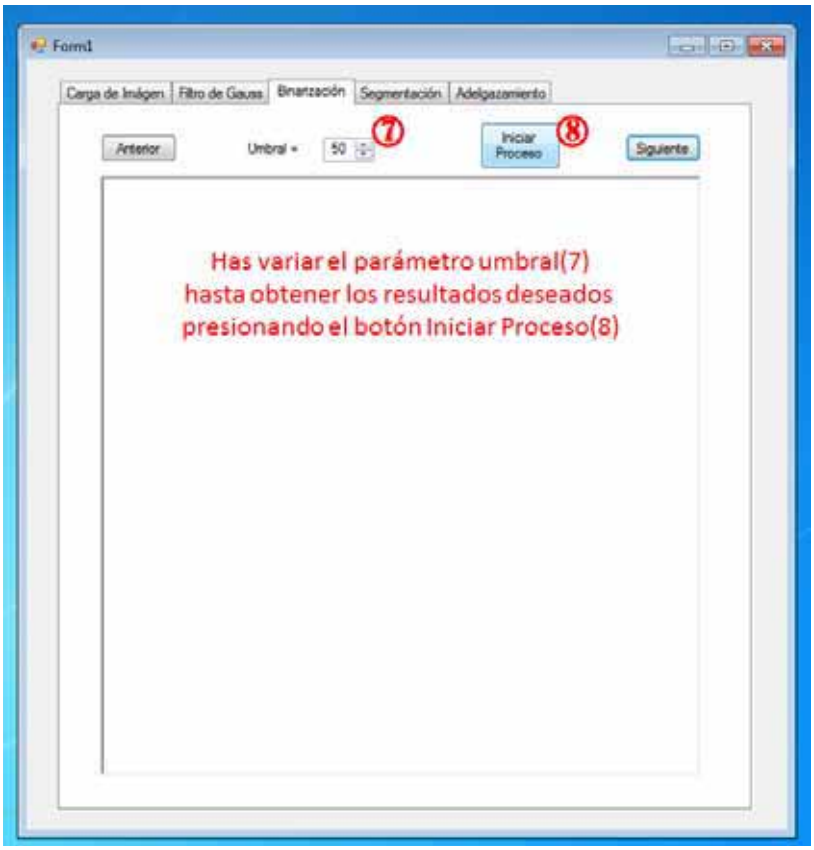
Guía rápida del producto

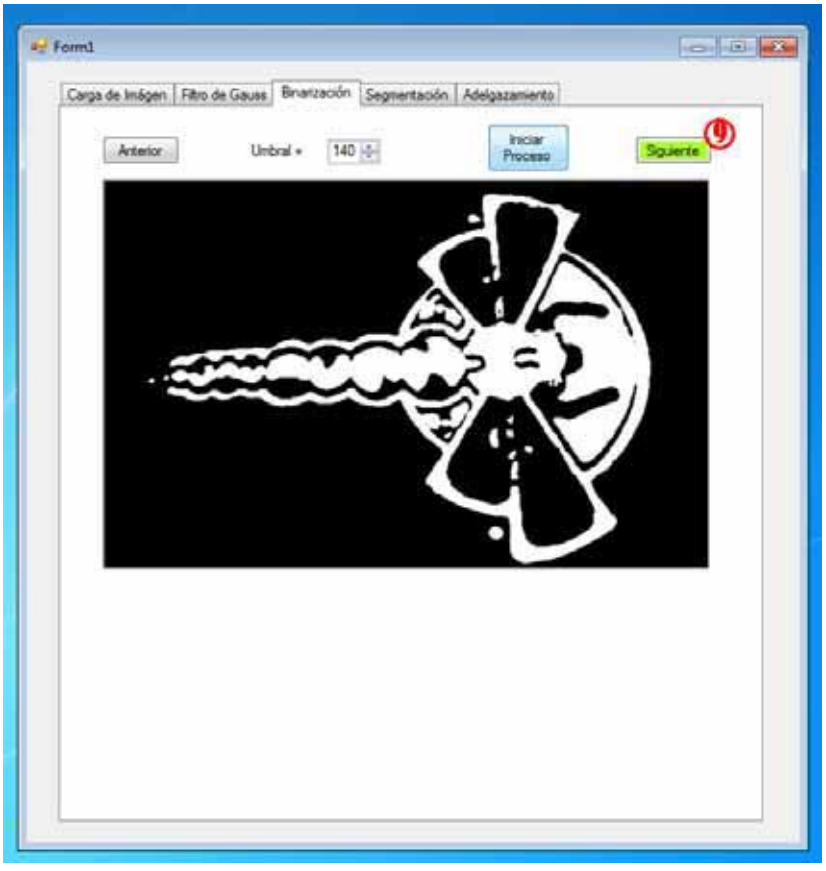


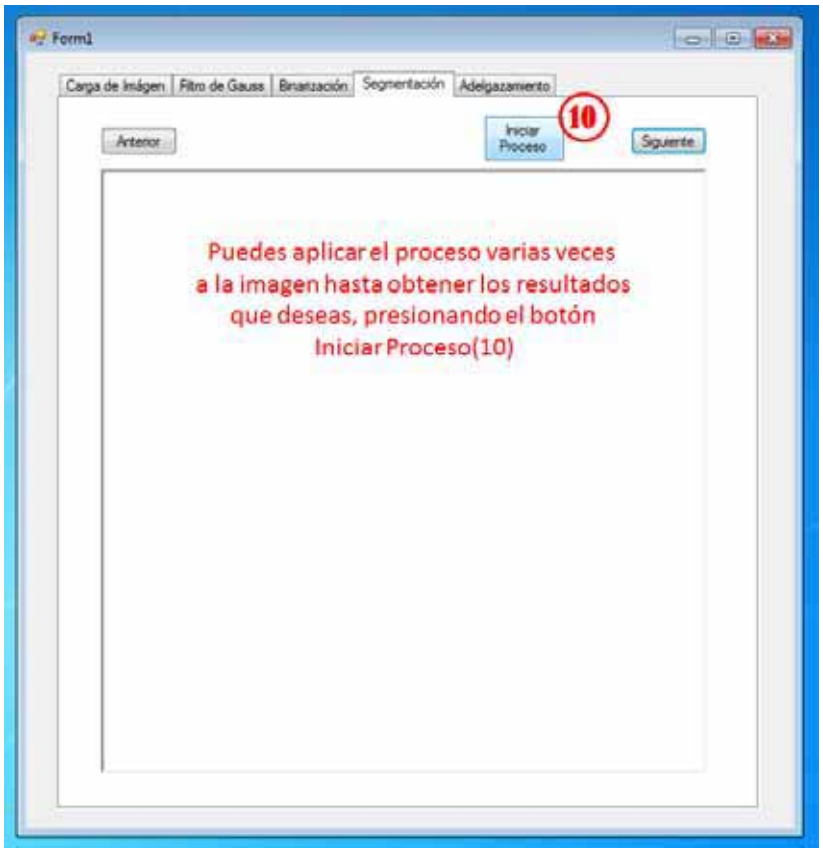


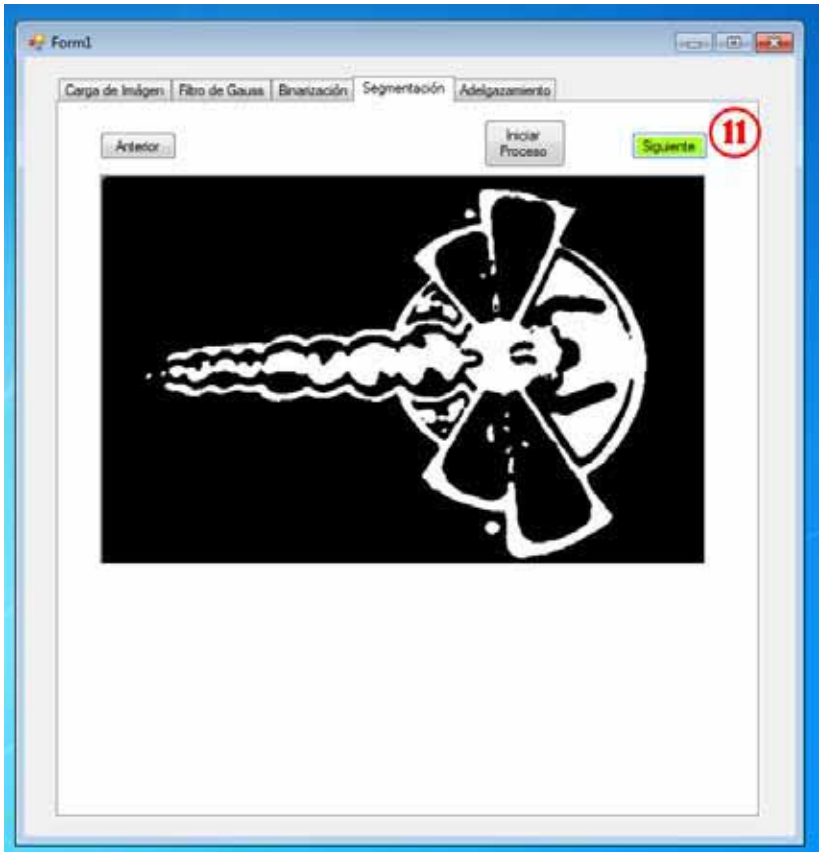

Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Sigma =
1.0
Radio =
3
Iniciar Proceso
Siguiente
3
4
5
Has variar los parámetros sigma(3) y radio(4)
hasta obtener los resultados deseados
presionando el botón Iniciar Proceso(5)


Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Sigma =
6.5
Radio =
9
Iniciar Proceso
Siguiente
6

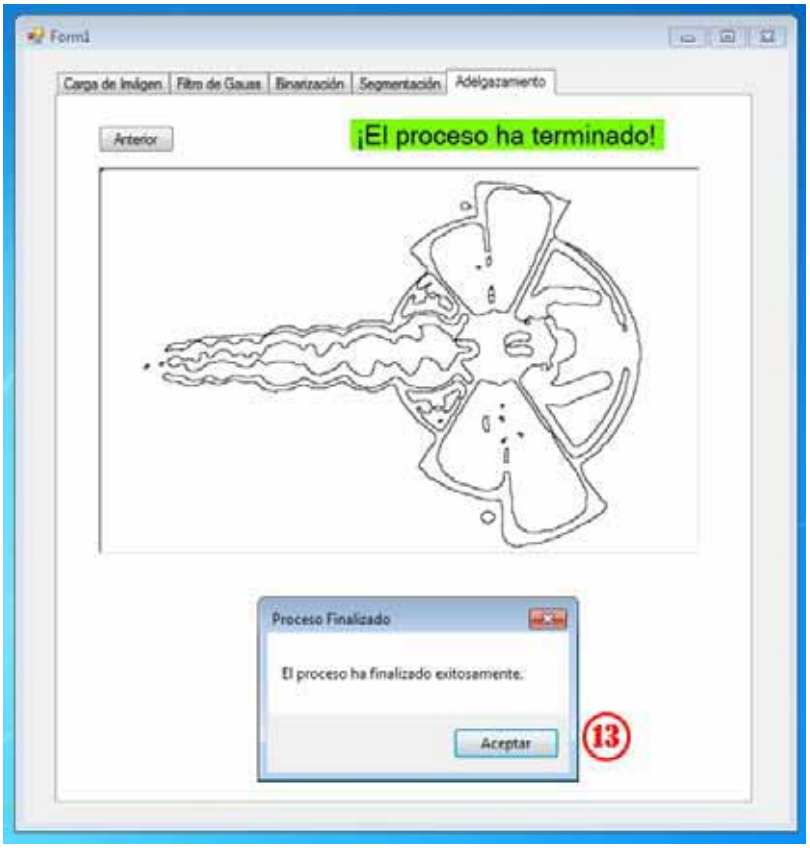

Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Umbral =
50
7
Iniciar Proceso
8
Siguiente
Has variar el parámetro umbral(7)
hasta obtener los resultados deseados
presionando el botón Iniciar Proceso(8)


Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Umbral =
140
Iniciar Proceso
Siguiente
9


Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Iniciar Proceso
10
Siguiente
Puedes aplicar el proceso varias veces
a la imagen hasta obtener los resultados
que deseas, presionando el botón
Iniciar Proceso(10)


Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Iniciar Proceso
Siguiente
11


Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Iniciar Proceso
12
Presionar el botón Iniciar Proceso(12)
para obtener el contorno de la figura.

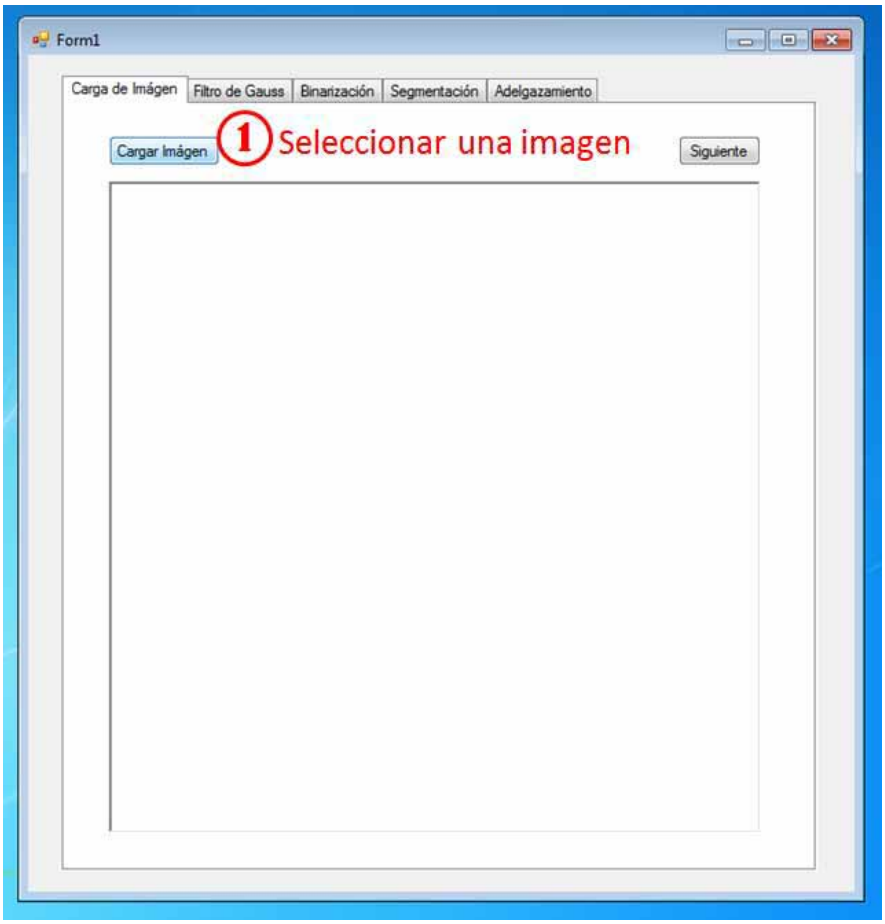

Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
¡El proceso ha terminado!
Proceso Finalizado
El proceso ha finalizado exitosamente.
Aceptar
13

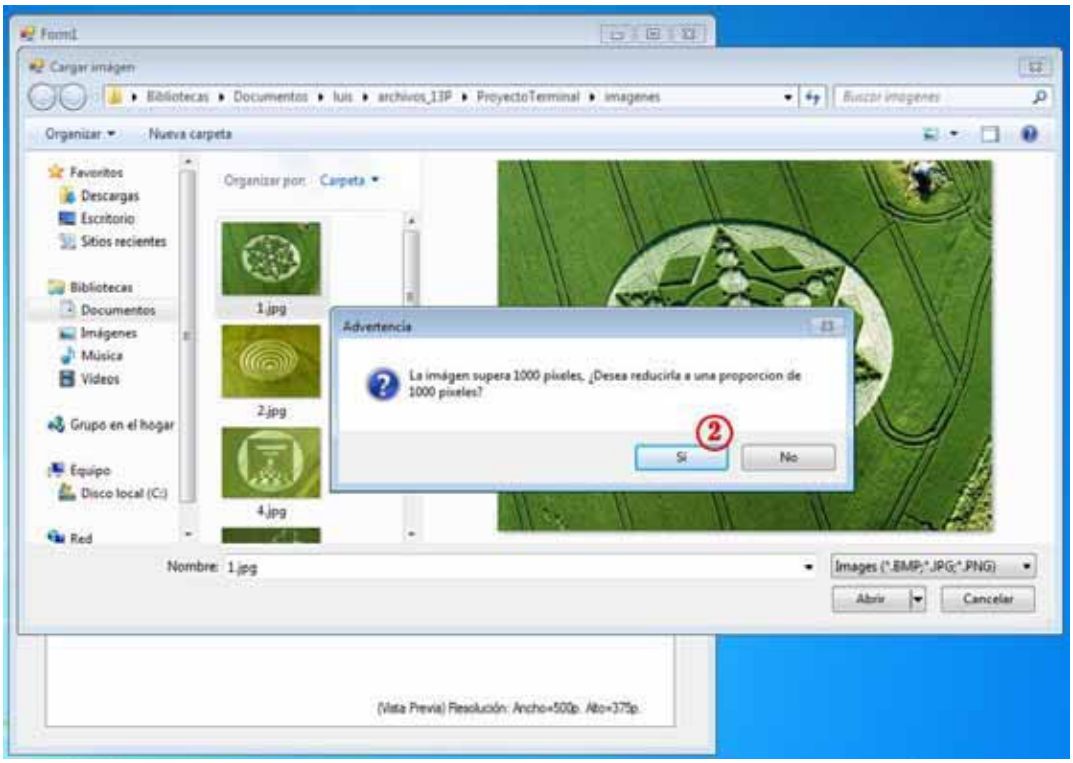

Ahora tus imágenes definitivas de cada etapa
del proceso están almacenadas en tu equipo,
ubicadas en la carpeta de instalación de la aplicación.

Manual de usuario

1. *Seleccionar imagen.* Lo primero que se tiene que hacer es cargar una imagen a la aplicación, considerando las siguientes especificaciones.

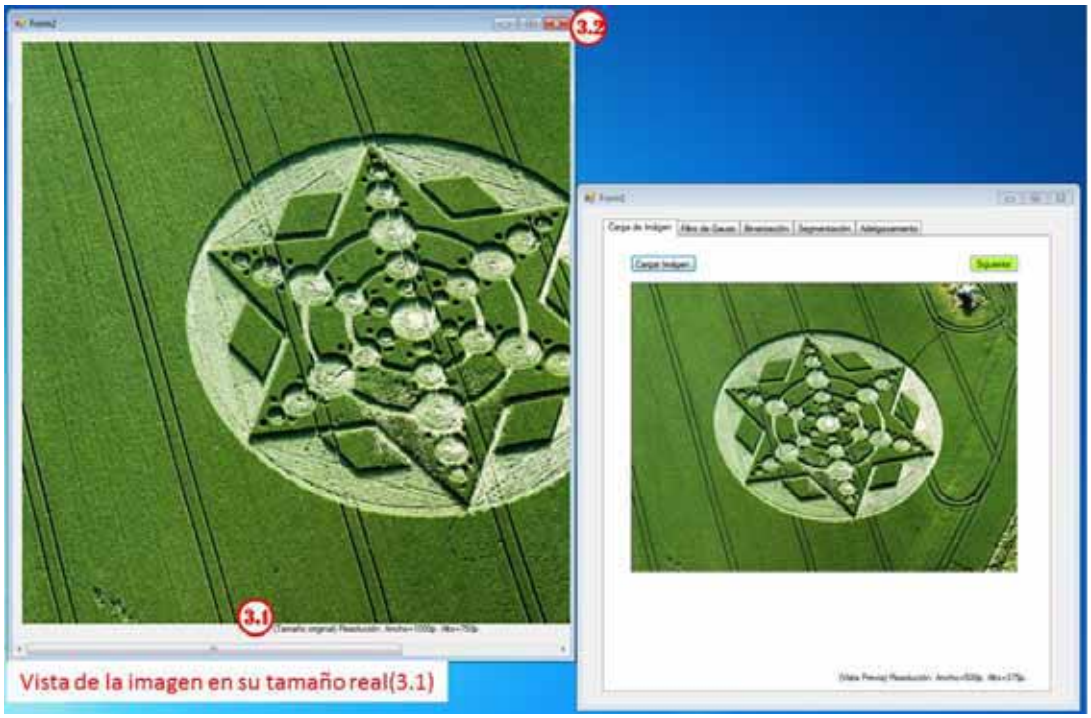
    - *Formato BMP, JPG y PNG.*
    - Tamaño en pixeles máximo de 1000 x 1000. (El programa incluye la opción de reducir el tamaño de la imagen para las que superen la especificación, sin modificar la imagen original). Ver paso (2). Si elige no usar nuestro método de reducción de imagen esta no se cargará en la aplicación.





La vista de la imagen en la aplicación siempre es en vista previa, podrá verla en tamaño real al dar click sobre la imagen. Ver puntos 3 y 4.

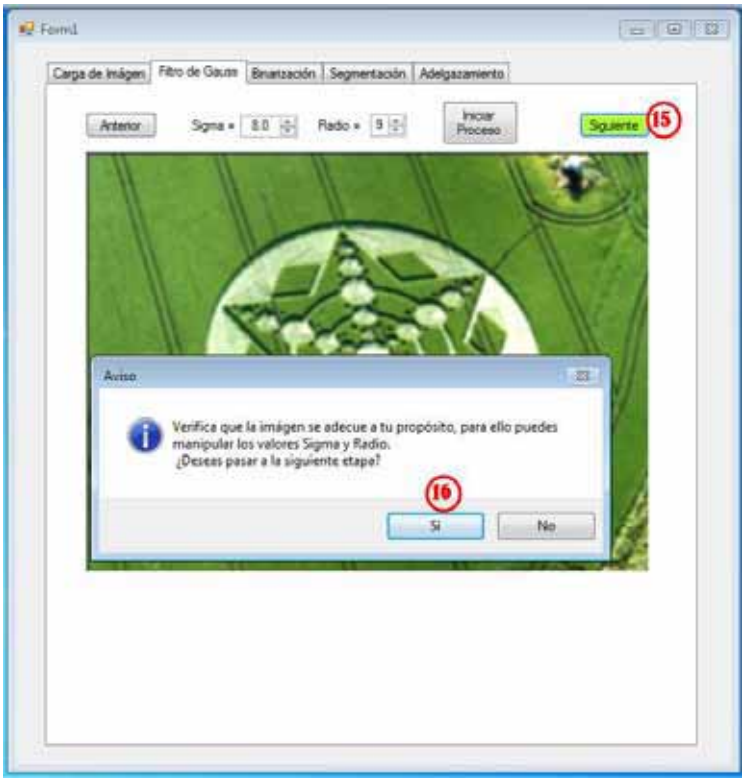
Cuando usted elija la opción para expandir la imagen a su tamaño real se abrirá otra pantalla donde se visualizará solo la imagen, recuerde cerrar la ventana en vista real. Ver puntos 3.1 y 3.2.

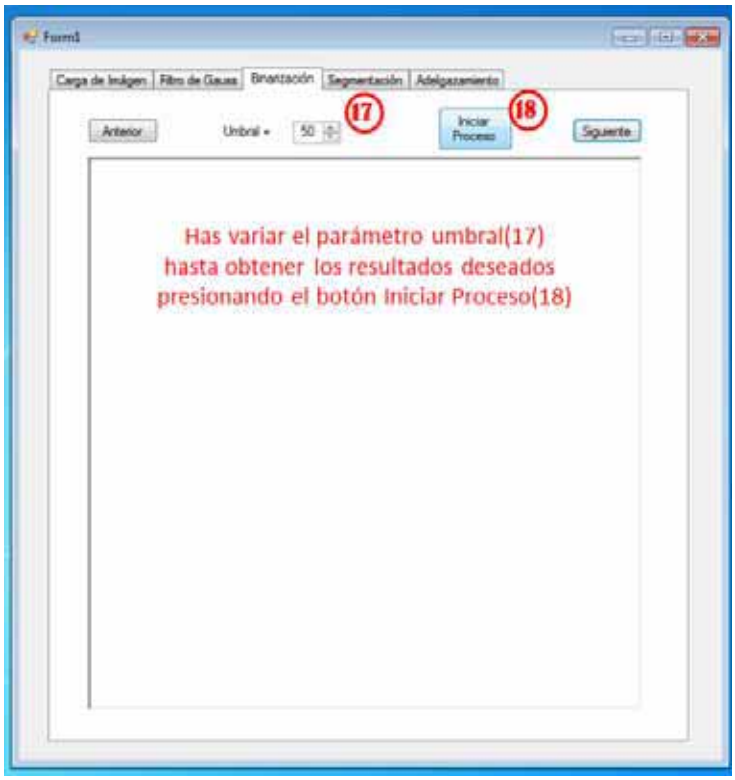


La aplicación controla que no se realicen etapas en diferente orden al especificado, así también cuando una etapa ya está lista se colorea en verde el botón de siguiente. Ver 5.

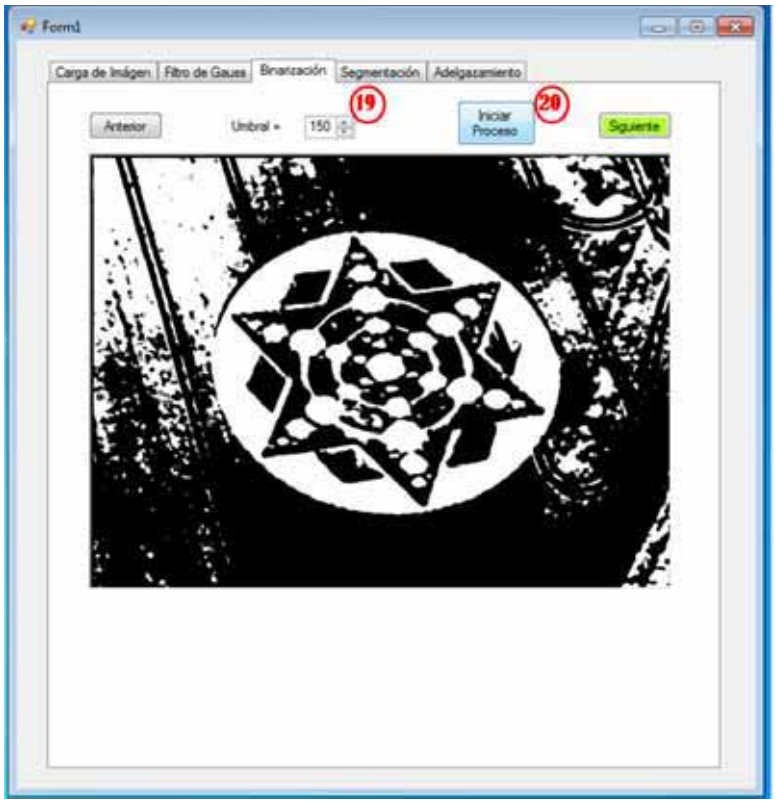
Se hacen variar los parámetros sigma y radio y se aplica el inicio del proceso hasta obtener los resultados que se deseen. Ver pasos 9 al 14.

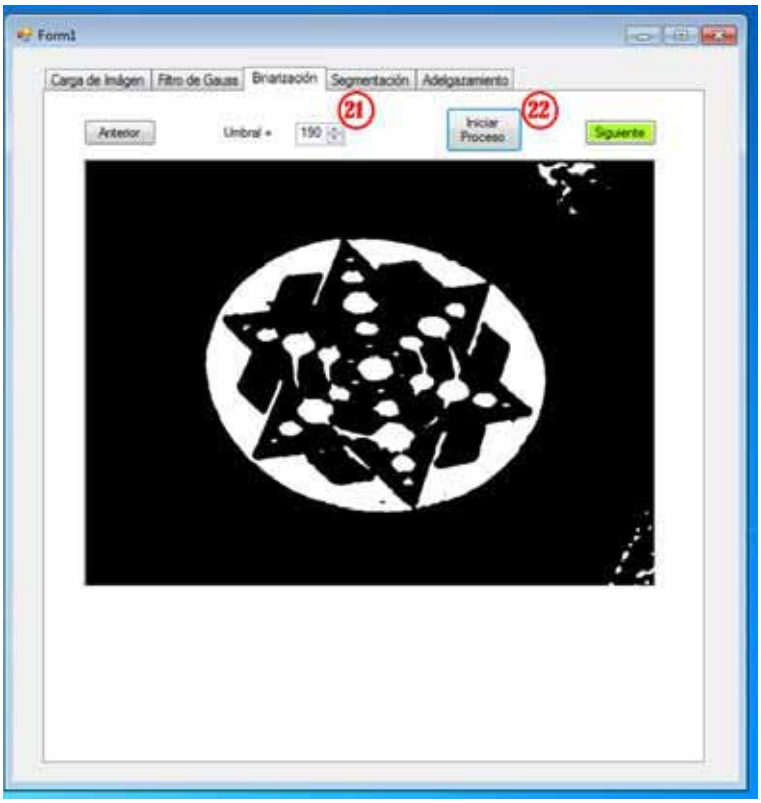
Una vez que estés seguro de que la imagen es la adecuada para cumplir el objetivo, la aplicación pedirá que confirme que está de acuerdo, si no es así la aplicación regresará a la etapa en proceso. Una vez comprobado avance a la siguiente etapa. Ver 15 y 16.



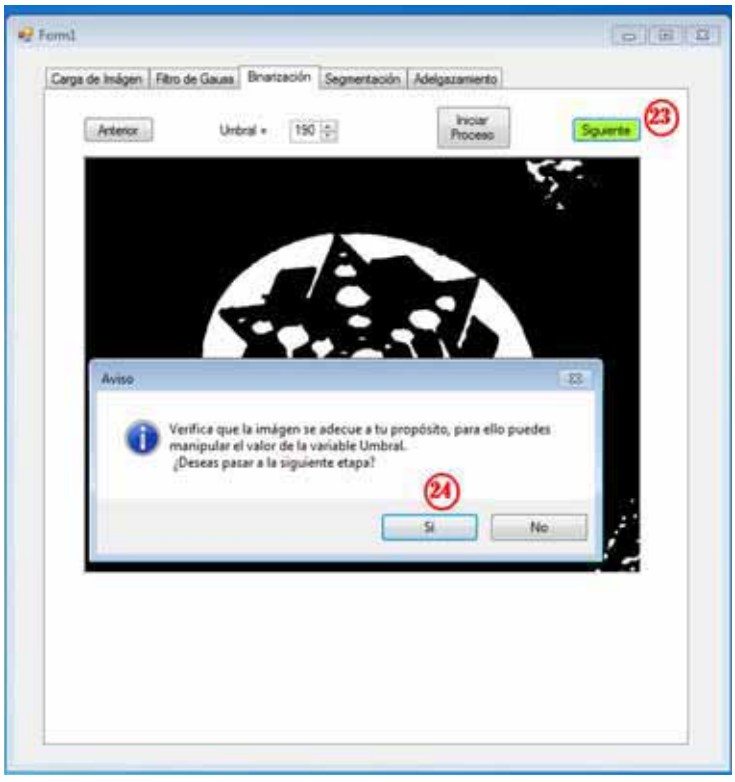


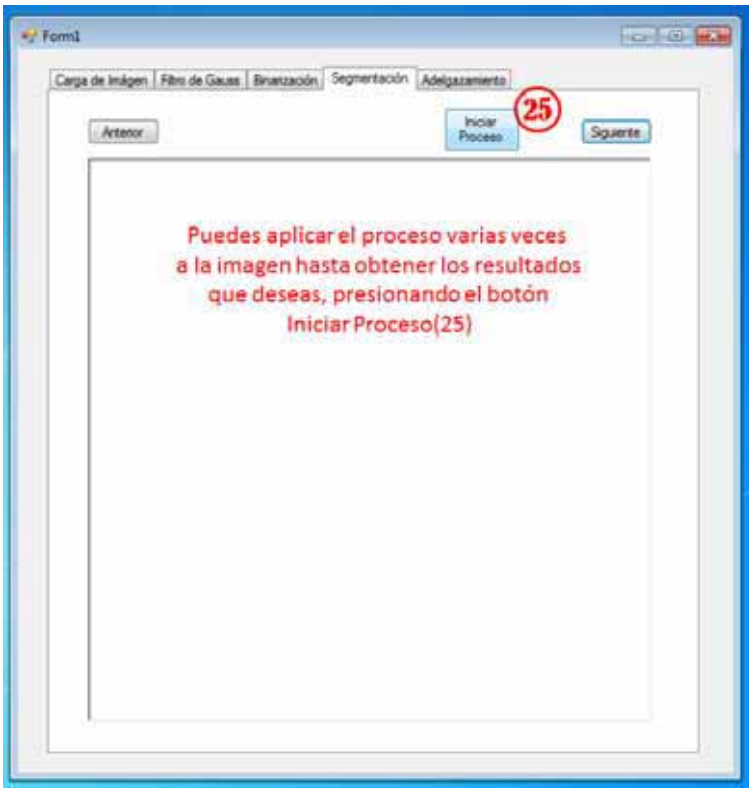
Se hace variar el parámetro umbral y se aplica el inicio del proceso hasta obtener los resultados que se deseen. Ver pasos 19 al 22.



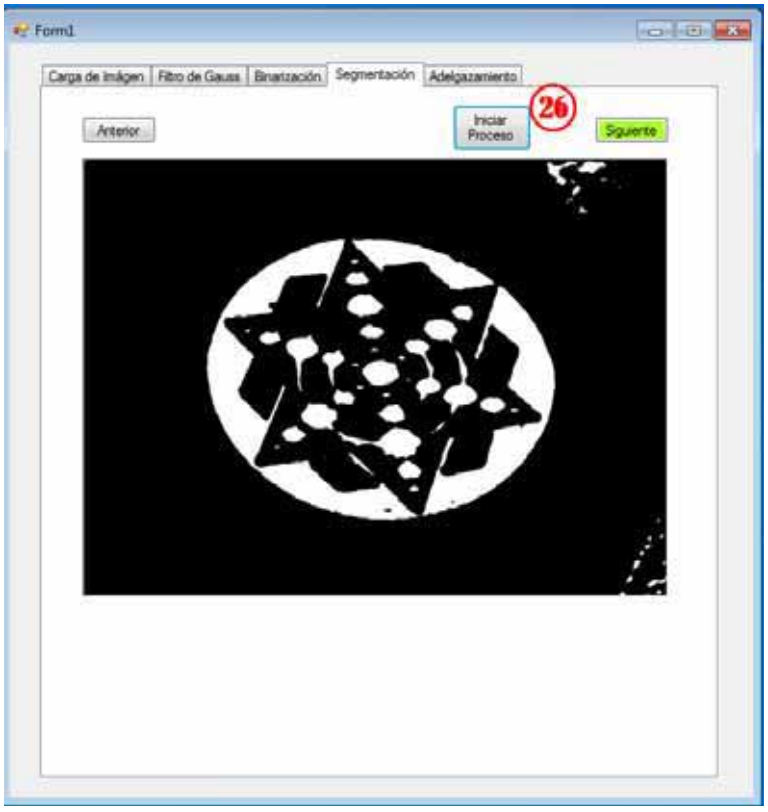


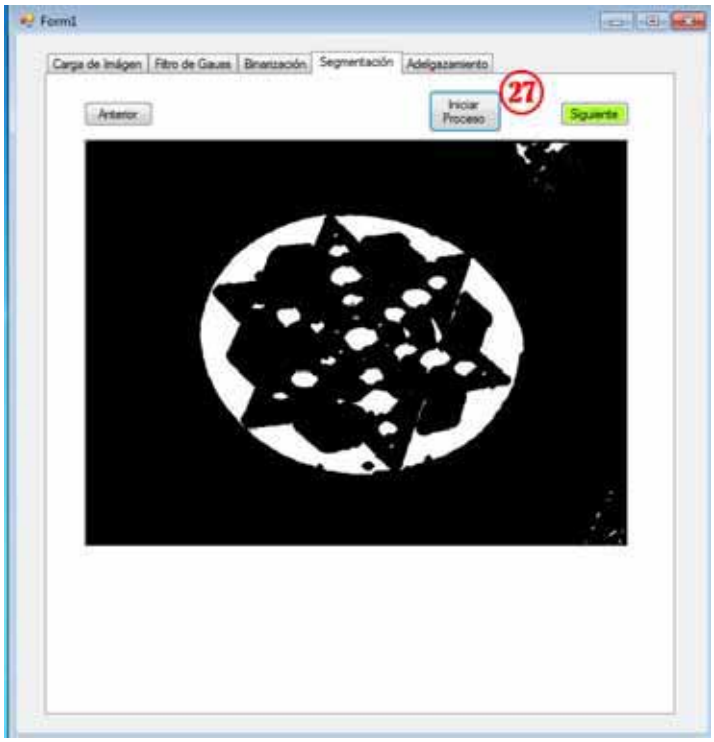
Una vez que estés seguro de que la imagen es la adecuada para cumplir el objetivo, la aplicación pedirá que confirme que está de acuerdo, si no es así la aplicación regresará a la etapa en proceso. Una vez comprobado avance a la siguiente etapa. Ver 23 y 24.



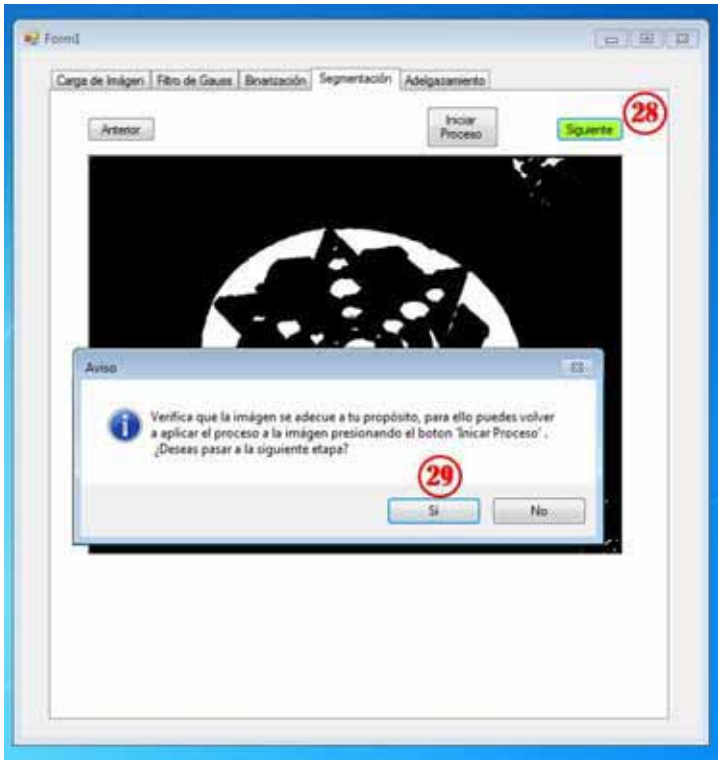


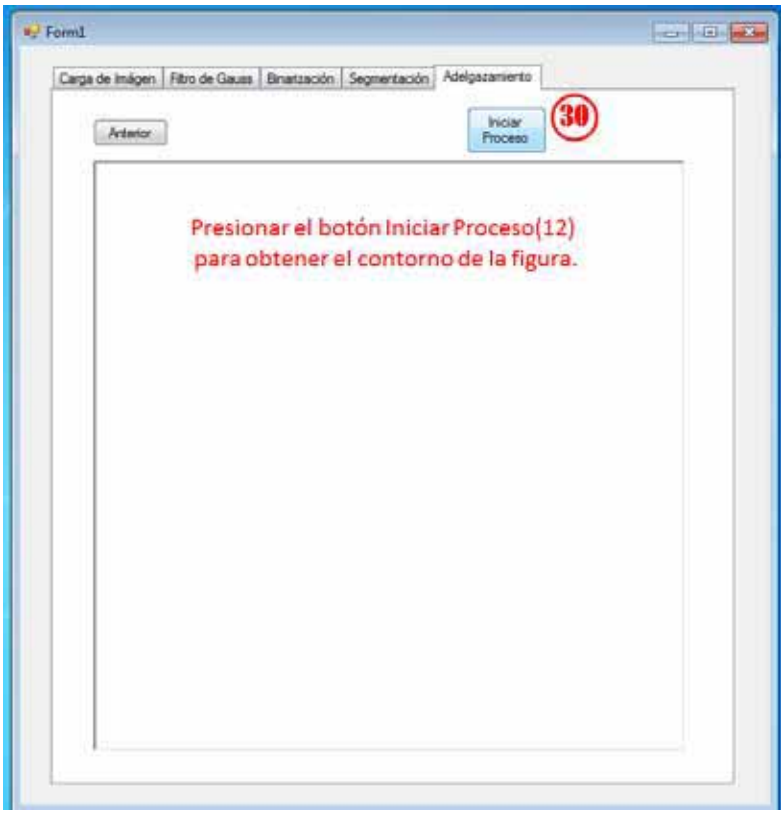
Se realiza varias veces el proceso hasta adecuar la imagen a lo requerido. Ver pasos 26 a 27.

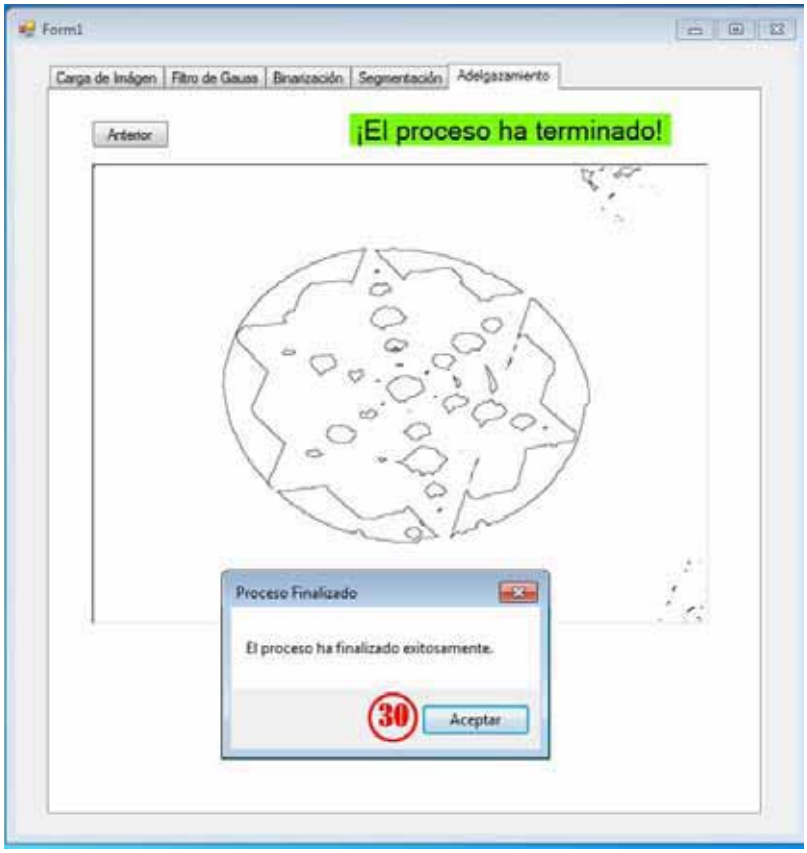




Una vez que estés seguro de que la imagen es la adecuada para cumplir el objetivo, la aplicación pedirá que confirme que está de acuerdo, si no es así la aplicación regresará a la etapa en proceso. Una vez comprobado avance a la siguiente etapa. Ver 28 y 29.




Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
Iniciar Proceso
30
Presionar el botón Iniciar Proceso(12)
para obtener el contorno de la figura.


Form1
Carga de Imágen
Filtro de Gauss
Binarización
Segmentación
Adelgazamiento
Anterior
¡El proceso ha terminado!
Proceso Finalizado
El proceso ha finalizado exitosamente.
30
Aceptar


Ahora tus imágenes definitivas de cada etapa
del proceso están almacenadas en tu equipo,
ubicadas en la carpeta de instalación de la aplicación.