

# Documentación del Código - Prueba 3

## Descripción General

El siguiente informe detalla el desarrollo de una aplicación de procesamiento de imágenes, creada como parte de la solución a la prueba 3 para la entrevista en el cargo de analista de ingeniería para la empresa Intecol. Esta aplicación permite cargar imágenes desde el sistema de archivos, aplicar filtros para mejorar la nitidez de la imagen y calcular la velocidad de un objeto en movimiento en la imagen.

## Explicación Detallada del Código

La aplicación está escrita en C# utilizando el entorno de desarrollo de Visual Studio. A continuación, se proporciona una explicación detallada de los componentes principales del código:

**Clase Form1:** Esta clase representa el formulario principal de la aplicación. Contiene métodos para cargar una imagen, procesar y calcular la velocidad del objeto en movimiento. Se utiliza la biblioteca OpenCvSharp para el procesamiento de imágenes.

**Método Cargar\_Button\_Click:** Este método maneja el evento de clic en el botón "Cargar". Permite al usuario seleccionar una imagen desde el sistema de archivos y la carga en la aplicación.

**Método Procesar\_Button\_Click:** Este método maneja el evento de clic en el botón "Procesar". Procesa la imagen cargada aplicando un filtro de aumento de nitidez y calcula la velocidad del objeto en movimiento en la imagen.

**Método ApplySharpen:** Este método aplica un filtro de aumento de nitidez a una imagen utilizando un kernel predefinido para realizar una operación de convolución.

**Método CalcularVelocidad:** Este método calcula la velocidad de un objeto en movimiento utilizando la longitud del objeto y el tiempo de exposición de la cámara.

## Algoritmos Utilizados

El algoritmo principal utilizado en la aplicación es el filtro de aumento de nitidez, implementado mediante una operación de convolución. Este filtro mejora la nitidez de la imagen resaltando los bordes y detalles importantes.

## Desarrollo

El sistema se desarrolló en el entorno de desarrollo Microsoft Visual Studio utilizando el lenguaje de programación C#. Se implementaron los siguientes pasos para el procesamiento de imágenes:

### Instalación de Paquetes NuGet:

Se requiere la instalación de los paquetes NuGet siguientes:

- OpenCvSharp4, OpenCvSharp4.Extensions, OpenCvSharp4.runtime.win, OpenCvSharp4.Windows y OpenCvSharp4.WpfExtensions para el procesamiento de imágenes.

### Carga de Imágenes:

- El usuario puede cargar la imagen del coche para el reconocimiento de la placa y velocidad

### Preprocesamiento de Imágenes:

- La imagen cargada es procesada implementando un filtro de nitidez para mejorar el reconocimiento de la placa

### Velocidad del coche:

- Usando los datos de longitud y tiempo de exposición de la imagen se calcula la velocidad del coche.

## Pruebas de Rendimiento

Se realizaron pruebas de rendimiento para evaluar la velocidad y eficacia de la aplicación en el procesamiento de imágenes. Estas pruebas incluyeron la carga de imágenes de diferentes tamaños y la medición del tiempo de procesamiento. La aplicación mostró un rendimiento satisfactorio, con tiempos de procesamiento aceptables incluso para imágenes de gran tamaño.

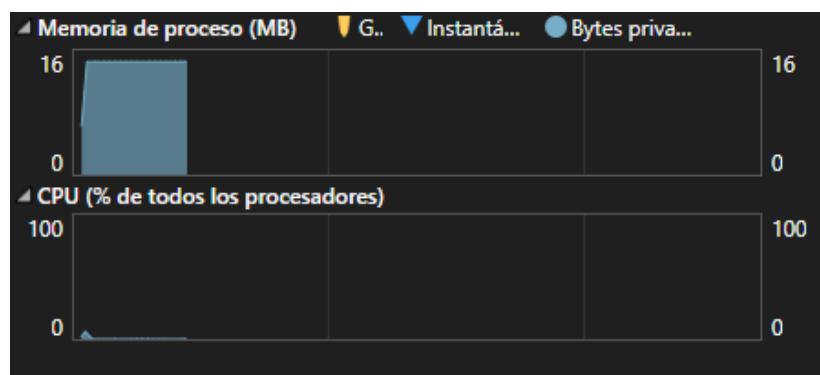


Figura 1: Consumo de memoria cuando el programa es ejecutado.

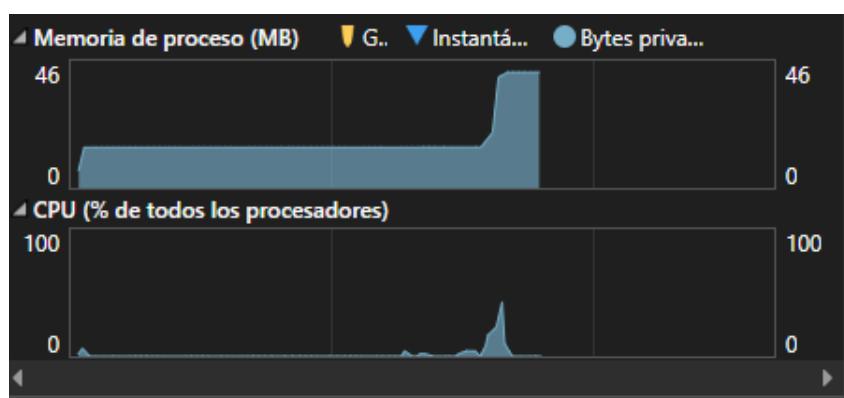


Figura 2: Consumo de memoria cuando el programa abre el explorador de archivos.

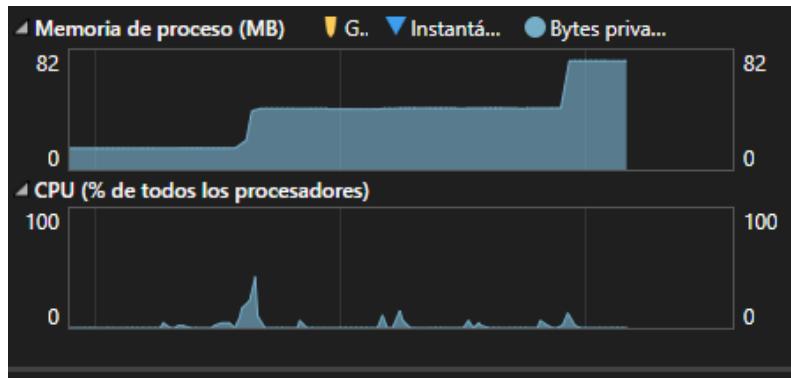


Figura 3: Consumo de memoria cuando el programa carga la imagen.

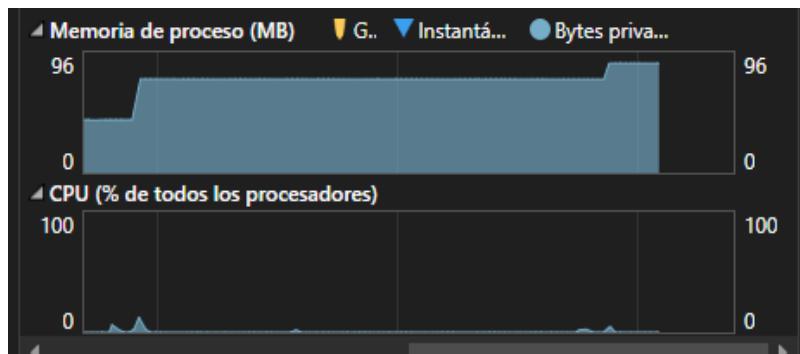


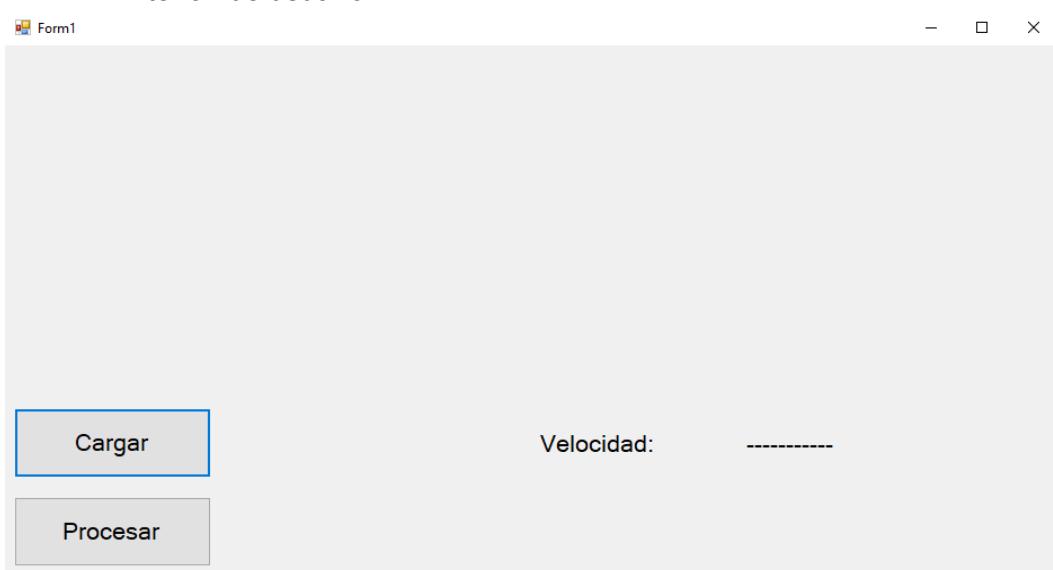
Figura 4: Consumo de memoria cuando la imagen es procesada.

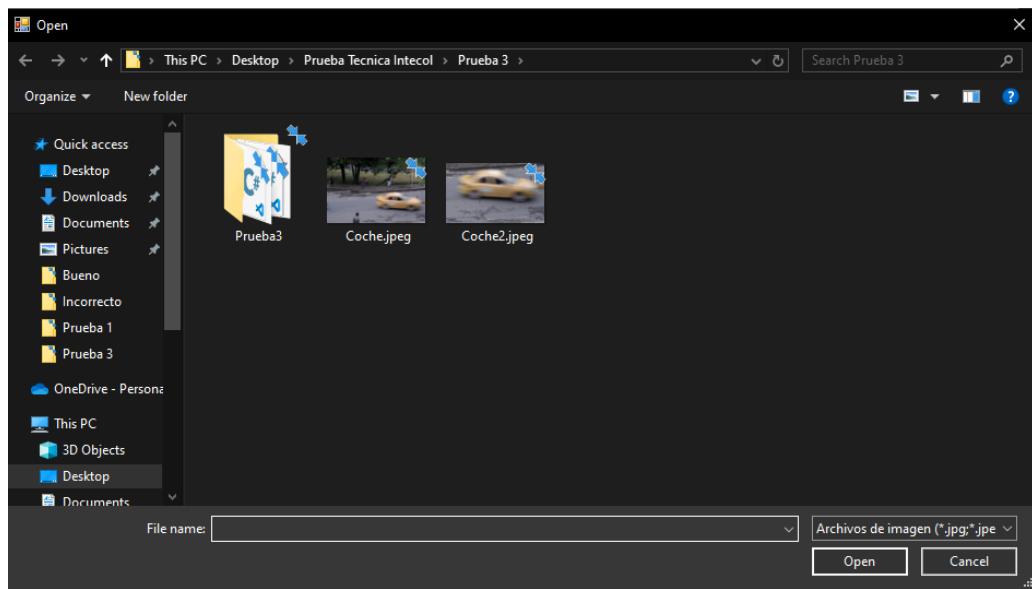
## Uso del Sistema

El uso del sistema es sencillo e intuitivo, y se puede seguir el siguiente paso a paso:

### Cargar una Imagen:

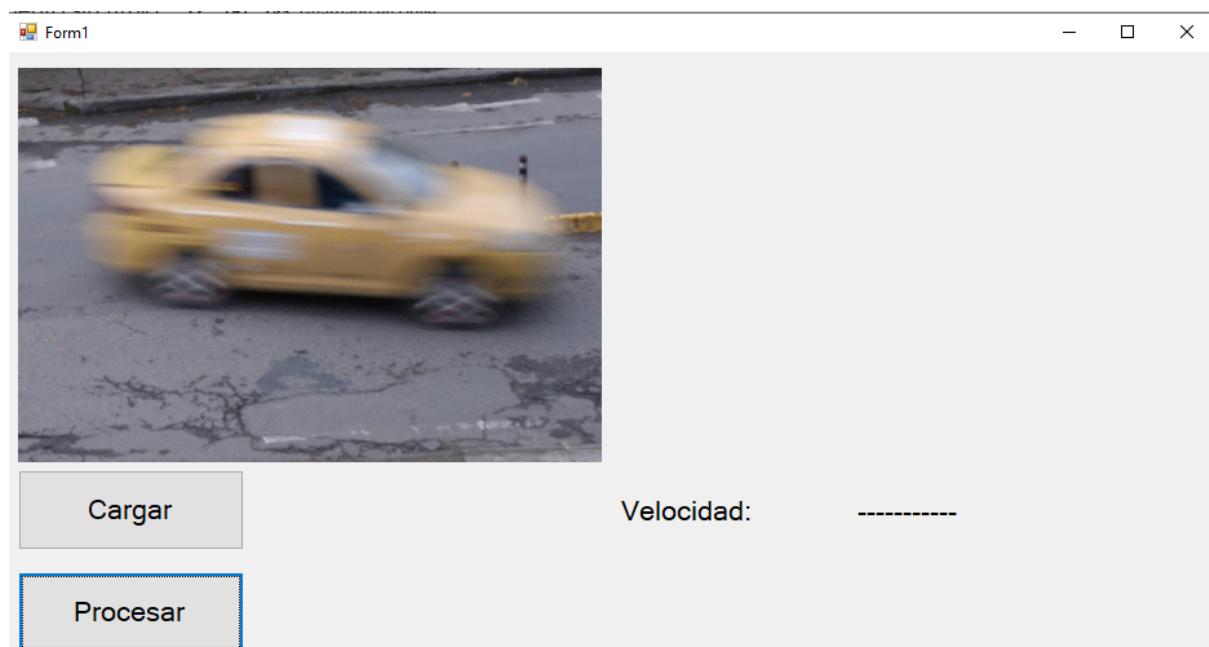
- Haga clic en el botón "Cargar".
- Se abrirá un cuadro de diálogo que le permitirá seleccionar la imagen que desea procesar desde su sistema de archivos.
- Una vez seleccionada la imagen, haga clic en "Abrir".
- La imagen cargada se mostrará en el PictureBox correspondiente en la interfaz de usuario.





### Procesar la Imagen:

- Despues de cargar la imagen, haga clic en el botón "Procesar".
- Se aplicará el filtro de desenfoque para mejorar la claridad de la imagen.
- La imagen procesada se mostrará en el otro PictureBox en la interfaz de usuario, permitiéndole compararla con la imagen original.





#### Visualizar la Velocidad del Objeto:

- Una vez procesada la imagen, la velocidad del objeto en movimiento se calculará automáticamente.
- La velocidad se mostrará en kilómetros por hora en el cuadro de texto correspondiente en la interfaz de usuario.

Velocidad: 18.00 km/h

```
C/C++  
/*  
 * Nombre del archivo: Form1.cs  
 * Descripción: Este archivo contiene la implementación de la clase Form1, que  
 representa la interfaz gráfica principal de la aplicación de procesamiento de  
 imágenes.  
 * Fecha de creación: 21/03/2024  
 * Autor: Ing. Santiago Alejandro Zuñiga Melo  
 * Descripción: Esta es mi solución a la prueba 2 para la entrevista en el cargo de  
 analista de ingeniería para la empresa Intecol.  
 *      La aplicacion permite cargar una imagen y enfocarla para observar mejor la  
 imagen y ademas poder observar la velocidad del coche  
 */  
  
using System;  
using System.Drawing;  
using System.Windows.Forms;
```

```
using OpenCvSharp;
using OpenCvSharp.Extensions;

namespace Prueba3
{
    /*
     * Clase: Form1
     * Descripción: Esta clase representa el formulario principal de la aplicación.
     Contiene métodos para cargar una imagen BMP,
     *      procesarla y realizar varias operaciones de procesamiento de imágenes,
     como aplicar filtros, detectar bordes
     *      y contar píxeles negros mediante segmentación.
    */
    public partial class Form1 : Form
    {
        private Mat loadedImage; // Variable para almacenar la imagen cargada
        private Mat processedImage; // Variable para almacenar la imagen procesada
        actual

        /*
         * Constructor: Form1
         * Descripción: Inicializa una nueva instancia de la clase Form1.
         *      Configura el DataGridView y el temporizador.
        */
        public Form1()
        {
            InitializeComponent();
        }
        /*
         * Método: Cargar_Button_Click
         * Descripción: Maneja el evento de clic en el botón "Cargar".
         *      Permite al usuario cargar una imagen desde el sistema de archivos.
        */
        private void Cargar_Button_Click(object sender, EventArgs e)
        {
            using (OpenFileDialog openFileDialog = new OpenFileDialog())
            {
                openFileDialog.Filter = "Archivos de imagen|*.jpg;*.jpeg;*.png;*.bmp";
                if (openFileDialog.ShowDialog() == DialogResult.OK)
                {
                    // Cargar la imagen utilizando OpenCvSharp
                    loadedImage = Cv2.ImRead(openFileDialog.FileName);

                    // Redimensionar la imagen al tamaño del PictureBox1
                    Mat resizedImage = new Mat();
                    Cv2.Resize(loadedImage, resizedImage, new
                    OpenCvSharp.Size(pictureBox1.Width, pictureBox1.Height));
                }
            }
        }
    }
}
```

```

        // Mostrar la imagen en PictureBox1
        pictureBox1.Image = resizedImage.ToBitmap();
    }
}
/*
 * Método: Procesar_Button_Click
 * Descripción: Maneja el evento de clic en el botón "Procesar".
 *      Procesa la imagen cargada y muestra el resultado en PictureBox2.
 *      También calcula y muestra la velocidad del objeto en la imagen.
 */
private void Procesar_Button_Click(object sender, EventArgs e)
{
    if (loadedImage != null)
    {

        // Aplicar un filtro de aumento de nitidez para enfocar la imagen
        processedImage = ApplySharpen(loadedImage);

        // Redimensionar la imagen procesada para mostrarla en PictureBox2
        Mat resizedImage = new Mat();
        Cv2.Resize(processedImage, resizedImage, new
        OpenCvSharp.Size(pictureBox2.Width, pictureBox2.Height));

        // Mostrar la imagen procesada en PictureBox2
        pictureBox2.Image = resizedImage.ToBitmap();

        // Calcular la velocidad del taxi
        double longitudPlaca = 50; // Longitud de la placa en centímetros
        double tiempoExposicion = 0.1; // Tiempo de exposición en segundos
        double velocidad = CalcularVelocidad(longitudPlaca, tiempoExposicion);

        // Mostrar la velocidad
        // Convertir la velocidad de cm/s a km/h
        double velocidadKmH = velocidad * 3600 / 100000;

        // Mostrar la velocidad en km/h
        VelocidadTXT.Text = $"{velocidadKmH.ToString("0.00")}" + " km/h";

        // Mostrar un MessageBox indicando que el procesamiento ha finalizado
        MessageBox.Show("Procesamiento completado.", "Información",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else
    {
        MessageBox.Show("Primero carga una imagen.", "Advertencia",
        MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
}

```

```
/*
 * Método: ApplySharpen
 * Descripción: Aplica un filtro de aumento de nitidez a una imagen.
 * Utiliza un kernel predefinido para realizar la operación de convolución.
 * Parámetros:
 * - image: La imagen a la que se aplicará el filtro.
 * Retorna:
 * - La imagen con el filtro aplicado.
 */
private Mat ApplySharpen(Mat image)
{
    // Crear el kernel para el filtro de aumento de nitidez
    Mat kernel = new Mat(3, 3, MatType.CV_32F, new float[] {
        -1, -1, -1,
        -1, 9, -1,
        -1, -1, -1
    });

    // Aplicar el filtro de convolución
    Mat sharpenedImage = new Mat();
    Cv2.Filter2D(image, sharpenedImage, -1, kernel);

    return sharpenedImage;
}
/*
 * Método: CalcularVelocidad
 * Descripción: Calcula la velocidad de un objeto en movimiento.
 * Parámetros:
 * - longitudPlaca: La longitud del objeto en movimiento (en centímetros).
 * - tiempoExposición: El tiempo de exposición de la cámara (en segundos).
 * Retorna:
 * - La velocidad del objeto (en cm/s).
 */
private double CalcularVelocidad(double longitudPlaca, double
tiempoExposición)
{
    // Calcular la velocidad utilizando la fórmula: velocidad = longitudPlaca / tiempoExposición
    return longitudPlaca / tiempoExposición;
}
}
```