Centro Nacional de Investigación y Desarrollo Tecnológico

Subdirección Académica

Departamento de Ciencias Computacionales

TESIS DE MAESTRÍA EN CIENCIAS

Implementación de algoritmos de Súper Resolución, en hardware dedicado, para imágenes de resonancia magnética

presentada por

Ing. Jaime Sacramento Pérez Gutiérrez

como requisito para la obtención del grado de
Maestro en Ciencias de la Computación

Directora de tesis
Dra. Andrea Magadán Salazar

Codirector de tesis
Dr. Raúl Pinto Elías

Cuernavaca, Morelos, México. Enero de 2018.

Cuernavaca, Morelos a 12 de enero del 2018
OFICIO No. DCC/014/2018

Asunto: Aceptación de documento de tesis

C. DR. GERARDO V. GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial del **Ing. Jaime Sacramento Pérez Gutiérrez**, con número de control M16CE014, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis profesional titulado **"Implementación de algoritmos de Súper Resolución, en hardware dedicado, para imágenes de resonancia magnética"** y hemos encontrado que se han realizado todas las correcciones y observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DIRECTOR DE TESIS

Dra. Andrea Magadán Salazar
Doctorado en Ciencias Computacionales
10654097

CO-DIRECTOR DE TESIS

Dr. Raul Pinto Elías
Doctor en Ciencias en la Especialidad de Ingeniería Eléctrica
3890453

REVISOR 1

Dr. Dante Mújica Vargas
Doctor en Comunicaciones y Electrónica
09131756

REVISOR 2

M.C. Gerardo Reyes Salgado
Maestro en Ciencias de la Computación
2493370

C.p. M.C. María Elena Gómez Torres - Jefa del Departamento de Servicios Escolares.
Estudiante
Expediente

NACS/lmz

Cuernavaca, Mor., 17 de enero de 2018
OFICIO No. SAC/061/2018

Asunto: Autorización de impresión de tesis

ING. JAIME SACRAMENTO PÉREZ
CANDIDATO AL GRADO DE MAESTRO EN CIENCIAS
DE LA COMPUTACIÓN
P R E S E N T E

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **"Implementación de algoritmos de Súper Resolución, en hardware dedicado, para imágenes de resonancia magnética"**, ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

ATENTAMENTE
"CONOCIMIENTO Y TECNOLOGÍA AL SERVICIO DE MÉXICO"

DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO

SEP TecNM
CENTRO NACIONAL
DE INVESTIGACIÓN
Y DESARROLLO
TECNOLÓGICO
SUBDIRECCIÓN
ACADÉMICA

C.p. M.T.I. María Elena Gómez Torres .- Jefa del Departamento de Servicios Escolares.
Expediente

GVGR/mcr

Dedicatoria

A mi madre, cuyo afectuoso amor e inagotable constancia ha hecho esto posible.

A mis hermanas, quienes siempre han sido la familia que quiero y necesito.

A mi cuñado, que me inspiró a seguir el camino de la ciencias computacionales y la tecnología.

Y a mi padre, cual rectitud, fuerza, valor e idealismo me motiva a ser mejor de lo que soy ahora.

"*La derrota no es motivo de vergüenza, si el espíritu sigue sin conquistar.*"

–Fenix, Starcraft

Agradecimientos

Quisiera agradecer al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado para financiar mis estudios de maestría.

A mi comité tutorial por la valiosa orientación a lo largo del desarrollo de esta tesis. Gracias, Dra. Andrea Magadán Salazar, Dr. Raúl Pinto Elías, Dr. Dante Mújica Vargas y Dr. Gerardo Reyes Salgado.

Al personal académico, administrativo, de mantenimiento y de seguridad del Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET), por el importante apoyo y servicio recibido durante mi estancia en Cuernavaca, Morelos.

A mi familia, quienes me apoyaron en momentos difíciles y gracias a ellos he logrado terminar esta meta.

A los nuevos amigos que he hecho en los dos años de posgrado y los viejos amigos que, como yo, decidieron probar su fortuna lejos de casa, viniendo a Morelos.

Por último, quiero agradecer al Dr. Madaín Pérez Patricio, el Dr. Hector Hernandez de León y al M.C Walter Torres Robledo del Instituto Tecnológico de Tuxtla Gutiérrez (ITTG) por sus consejos e instrucción que me llevaron a estudiar un posgrado.

Muchas gracias a todos, soy afortunado de tener a tantas personas con quienes estar agradecido.

Resumen

En este documento de tesis se propone la implementación de algoritmos clásicos de súper resolución (SR) en un sistema de hardware dedicado de alto rendimiento para mejorar la resolución de las imágenes de resonancia magnética.

Después de un análisis del estado del arte, se seleccionaron tres algoritmos clásicos de Súper Resolución: el vecino más cercano, interpolación bilineal e interpolación bicúbica. Se realizaron pruebas usando evaluaciones propuestas para este trabajo y determinar el algoritmo más eficiente en el procesamiento de imágenes médicas utilizando cuatro estudios de fuentes privadas y públicas conformadas por más de 300 imágenes cada una.

Para la implementación de los algoritmos seleccionados, se utilizó un sistema embebido con una tarjeta gráfica integrada para aprovechar un incremento en el velocidad del procesamiento usando las características del cómputo paralelo.

Los resultados obtenidos mostraron que el desempeño de la interpolación bilineal fue superior a la alcanzada por los otros dos algoritmos (en función del método de evaluación estadístico propuesto), el cual mostró mejores resultados con respecto a la calidad de la imagen y el tiempo de ejecución.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este capítulo se presenta la introducción al tema que aborda el proyecto de tesis, las imágenes de resonancia magnética, problemática a tratar, los objetivos del proyecto, los alcances y limitaciones, el método de solución propuesto y por último, la organización por la cual está constituida esta tesis.

1.1. Introducción

Los pixeles son las unidades mínimas con las cuales se constituyen las imágenes digitales (Gonzales y Batchelor, 1978). Su nombre proviene del acrónimo en inglés *picture element* y generalmente llegan a representarse con distintas tonalidades de color: blanco y negro, gris y el sistema RGB (Red Green Blue).

El concepto de resolución en una imagen se refiere a la densidad de pixeles que contiene. El término de resolución también suele ser definido tanto por la anchura y altura de una imagen. Haciendo referencia a un ejemplo, una imagen que tenga 512 pixeles de ancho y 1024 pixeles de alto (512 x 1024) contiene una resolución de 524,288 pixeles.

La resolución de una imagen es un factor crítico para lograr mejor reconocimiento visual, tanto para los humanos como en las aplicaciones de visión por computadora. Una imagen con una alta resolución es siempre deseable. La resolución es importante para diversas áreas tales como: imágenes médicas, vídeo vigilancia, entretenimiento, astronomía, entre otros. Para mostrar un ejemplo, la Figura 1.1 presenta la diferencia visual mediante el acercamiento (zoom) de una región en la imagen (a) con la imagen (b) que tiene la mitad de resolución en pixeles.

(a) Imagen original

(b) Versión con menor resolución

Figura 1.1: Diferencia entre dos imágenes de distinta resolución.

Las pequeñas diferencias tienden a contener detalles importantes en la extracción del conocimiento representado en las imágenes. En imágenes médicas una resolución alta conllevará a una mejor segmentación o clasificación en las regiones o aumentará la precisión con la que se localizan los cuerpos extraños como los tumores (Isaac y Kulkarni, 2015).

Para aumentar la resolución de las imágenes existen técnicas de procesamiento de señales cuya función es aumentar la calidad representativa en las imágenes. Dichas técnicas son denominadas como: súper resolución (SR) (Salvador, 2017). El objetivo de estos algoritmos llega a definirse de la siguiente forma: Dada una entrada visual de baja resolución (ya sea una imagen de baja resolución o un conjunto de imágenes, por ejemplo, que correspondan a la secuencia de un vídeo), se estima una salida correspondiente de alta resolución.

En la literatura se han trabajado con varias técnicas de súper resolución (SR), las cuales han sido objeto de estudio por más de tres décadas. (Yue, 2016). Los métodos de SR suelen requerir equipos costosos para su implementación, en especial con imágenes muy grandes. Estos equipos tienen la desventajas de tener poca o nula portabilidad; es tedioso de transportar y requiere ser proveída de una alimentación eléctrica de corriente alterna, la cuál puede no estar siempre disponible.

En imágenes médicas, una imagen que posea una resolución alta siempre es deseada por los especialistas médicos. Sin embargo, existen algunas problemáticas, como lo es el proceso de adquisición: es tedioso para el paciente debido a los largos lapsos a los que debe ser expuesto, lo cual trae como consecuencia que tienda a moverse, provocando la aparición de artefactos en las imágenes y reduciendo su calidad visual. Ademas, el equipo de cómputo que procesa las señales de la máquina de resonancia es costoso y pesado, haciendo difícil su manejo.

Por lo tanto, el propósito de este proyecto de tesis es: **implementar y evaluar tres algoritmo de SR en un sistema embebido de alto rendimiento para el procesamiento de imágenes médicas en resonancia magnética, con el fin de reducir la experiencia perjudicial al paciente y mejorar la portabilidad del equipo de cómputo.**

1.2. Imágenes de resonancia magnética

Las imágenes de resonancia magnética son representaciones visuales del cuerpo humano, creadas mediante radiación no ionizada, con el propósito de servir de herramientas de apoyo en los diagnós-

ticos médicos (Pooley, 2005). El proceso de adquisición utiliza señales de radio para ser emitidas en el cuerpo humano y recibirlas de vuelta con información de la estructura interna como son órganos, huesos y cartílagos. Tal información es mandada a una computadora para ser transformada en imágenes digitales. El uso de imágenes de resonancia magnética tiene ciertas ventajas como son:

- Visualización interna del cuerpo sin usar radiación ionizada (rayos X).
- Adquisición de múltiples planos sin mover al paciente.
- Tienen mejor contraste en las partes blandas que las tomografías.

Pero, también existen ciertas desventajas que son:

- Costo muy elevado.
- Tiempos de adquisición largos.
- Tendencia de aparición de anomalías en la imagen.
- Peligro con objetos susceptivos a la atracción magnética, como lo son los metales.

Dadas estas ventajas y desventajas, las imágenes de resonancia magnéticas presentan ser casos de estudio interesantes, tanto para el área de la medicina como el procesamiento de imágenes digitales.

1.3. Problemática

Existen deficiencias en los momentos de adquisición de los datos crudos de una imagen. Tales imágenes llegan a presentar desperfectos provenientes de los mismos sensores de adquisición.

Las máquinas de resonancia magnética requieren largos periodos de adquisición para obtener imágenes con una alta resolución. Esto provoca que los pacientes experimenten extensos tiempos de tratamiento para sus estudios, lo cual tiene como consecuencia que sufran de claustrofobia y no puedan permanecer quietos, resultando en distorsiones en las imágenes.

Por ende, se propone, para este proyecto de tesis, aminorar las siguientes problemáticas:

1. Reducir los tiempos de adquisición sin comprometer la resolución de las imágenes: las imágenes de resonancia magnética obtienen una calidad de resolución en función del tiempo de muestreo que tengan. Se propone reducir el tiempo mediante la mejora en la resolución de las imágenes, después de la adquisición.
2. Mejorar la experiencia del proceso para el paciente: los largos procesos de adquisición provocan efectos negativos tanto para el paciente como para el proceso. Reducir los tiempos significa aumentar la comodidad del paciente.

1.4. Objetivos

Los objetivos planteados para este proyecto de tesis se presentan a continuación, describiendo al objetivo general y a los específicos.

1.4.1. Objetivo general

Implementar un sistema de procesamiento de imágenes basado en algoritmos clásicos de súper resolución para la manipulación de imágenes de resonancia magnética en un sistema embebido de alto rendimiento.

1.4.2. Objetivos específicos

1. Estudiar tres algoritmos clásicos de SR.
2. Evaluar dichos algoritmos con métodos cuantitativos para determinar cual es el mejor.
3. Desarrollar un sistema de procesamiento de imágenes con el algoritmo seleccionado para MRI(Magnetic Resonance images).
4. Implementar el sistema en una tarjeta de hardware dedicado de alto rendimiento.

1.5. Alcances y limitaciones

Para simplificar la investigación se propusieron los siguientes alcances y limitaciones:

1.5.1. Alcances

- Evaluar el desempeño de al menos tres técnicas de SR clásicas en MRI.
- Trabajar con imágenes de resonancia magnética que proporcionen vistas del cerebro, columna y las rodillas.
- Implementación del sistema, junto con el algoritmo seleccionado, en el hardware dedicado de alto rendimiento.

1.5.2. Limitaciones

- Utilizar Opencv como apoyo en la implementación del sistema de procesamiento.
- Solo se realizará el procesamiento de las imágenes y no se incluirá algún tipo de detección o extracción del conocimiento.

1.6. Método de solución

Para la metodología de solución de este proyecto de tesis se elaboró un plan que consta de seis fases: Fundamentación, Especificación, Desarrollo, Integración, Evaluación y finalmente Documentación.

1. **Fundamentación:** Búsqueda del estado del arte con respecto a trabajos previos relacionados.

2. **Especificación:** Análisis y diseño de la solución al problema junto con la selección de las técnicas a utilizar. También consta de la selección de las herramientas de trabajo; lenguaje de programación y plataforma de hardware.

3. **Desarrollo:** Codificación del sistema de visión para la manipulación de las imágenes médicas y del sistema de procesamiento de imágenes para la plataforma de hardware seleccionada.

4. **Implementación:** Implementación del sistema procesamiento de imágenes en el hardware dedicado.

5. **Evaluación:** Realización de pruebas funcionales del sistema implementado y verificación de la terminación en los objetivos impuestos.

6. **Documentación:** Redacción del documento de tesis y artículo técnico con los resultados obtenidos del trabajo.

1.7. Organización de la tesis

El presente documento de tesis se organiza de la siguiente forma:

- **Estado del arte:** En el Capítulo 2 se presenta la extensa investigación realizada en la literatura con la cuál se ha fundamentado la base teórica y práctica de este proyecto. Se incluyen aportaciones de SR tanto en el área médica como en propósitos generales, trabajos hechos con la plataforma de cómputo paralelo CUDA, aportaciones con métricas para evaluaciones cuantitativas, sistemas embebidos y por último, el análisis del estado del arte.

- **Algoritmos de Súper Resolución y métricas de evaluación:** en el Capítulo 3 se presenta la introducción a la importancia y el concepto de la súper resolución, la complejidad asintomática de los algoritmos, los tres algoritmos de SR seleccionados y por último, las métricas de evaluación cuantitativas.

- **Implementación en el sistema embebido Jetson TK1:** En el Capítulo 4 se presenta una introducción al contexto relacionado con el cómputo paralel,. un breve resumen de la plataforma de cómputo paralelo CUDA (Computer Unified Device Architecture), la tarjeta de desarrollo Jetson TK1, la librería de visión por computadora OpenCV4Tegra, computación paralela en GPU, la configuración del sistema embebido y por último, la configuración de la GPU para su implementación en este trabajo.

- **Experimentación y análisis de resultados:** en el Capítulo 5 se presenta las bases de datos utilizadas, la metodología de experimentación, el entorno de pruebas, los resultados de las evaluaciones en CPU y GPU, el análisis de los resultados y por último, la comprobación de los resultados obtenidos

- **Conclusiones:** en el Capítulo 6 se presenta una introducción redactando, en resumen, lo realizado para este proyecto de tesis, la conclusión general, los objetivos completados, las aportaciones del trabajo y por último, los trabajos futuros.

Capítulo 2

Estado del arte

En este capítulo se presenta el estado del arte dividido en cuatro partes: súper resolución en imágenes médicas, súper resolución en otras aplicaciones, CUDA, métricas para evaluaciones cuantitativas, sistemas embebidos y por último análisis del estado del arte

2.1. Introducción

La clasificación de las técnicas en SR varían a lo largo de los trabajos realizados en función de los métodos existentes que van surgiendo. De acuerdo con (Isaac y Kulkarni, 2015) los algoritmos se clasifican mediante dos aproximaciones: las que utilizan múltiples imágenes y solo una. Los métodos de SR basados en múltiples imágenes hacen uso de las secuencias que existen en estas con el fin de extraer y combinar los datos complementarios mediante algoritmos del dominio espacial para realizar el proceso de reconstrucción. Por otro lado, los métodos de una sola imagen hacen un procedimiento similar con la diferencia de que suelen utilizar métodos de reconstrucción basados en técnicas de interpolación. Otros hacen uso de bases de datos externas en las imágenes a trabajar junto con técnicas de *machine learning* en la extracción de la información.

Ambas aproximaciones se han aplicado en variados campos dependiendo de las problemáticas encontrada, por ejemplo, en vídeo vigilancia las técnicas de múltiples imágenes son extensamente usadas para mejorar la detección de objetos en tiempo real. Usualmente, los métodos de una sola imagen han sido trabajados extensamente en imágenes médicas debido al insuficiente muestreo con respecto a la toma de una escena (Isaac y Kulkarni, 2015).

En este documento de tesis el estado del arte se organizó de acuerdo al objetivo del trabajo; es decir, su aplicación en imágenes médicas y su implementación en sistemas embebidos.

2.2. Súper Resolución en imágenes médicas

En esta sección se exponen los trabajos de SR relacionados en las áreas médicas.

2.2.1. Reconstrucción en imágenes de resonancia magnética mediante Súper Resolución con k-t SPARSE-SENSE en su núcleo

En este trabajo (Malczewski, 2013), los autores presentan como objetivo combinar los algoritmos de SR con el *framework* de censado compreso (CC) para obtener MRI de alta resolución. Tiene como aportación presentar un nuevo algoritmo de SR que aproveche la escasez de datos en las MRI para reducir el tiempo de adquisición y mejorar su resolución.

La metodología del algoritmo es la siguiente:

1. Se estiman los valores de la imagen en alta resolución.
2. El proceso de reconstrucción comienza simulando un conjunto de imágenes en baja resolución que correspondan a la imagen de entrada.
3. Las imágenes de baja resolución se usan para mejorar la versión original de la imagen en a una de más alta resolución mediante la retropropagación de cada valor de pixel en las imágenes de baja resolución simuladas.
4. Se repite el proceso hasta obtener la imagen de alta resolución con el menor índice de error posible.

Los resultados fueron evaluados de manera cualitativa usando la comparación visual en las imágenes reconstruidas, llegando a conclusiones subjetivas.

Comentarios: el artículo revela un interesante método para el proceso de reconstrucción, donde se utilizaron imágenes de baja resolución simuladas en lugar de adquirirlas desde el mismo dispositivo.

2.2.2. Una representación bayesiana escasa para súper resolución en imágenes de resonancia magnética cardíacas

En este trabajo (Velasco, Rueda, Santa Marta, y Romero, 2017), los autores presentan como objetivo mejorar las imágenes cardíacas mediante un método de SR utilizando aproximación bayesiana de datos escasos. La aportación es un método de SR basado en una representación bayesiana que es capaz de lidiar con la incertidumbre de múltiples fuentes de ruido o con voxels no ortogonales.

Para la metodología se propone tomar N imágenes de baja resolución que corresponden a una serie de dos dimensiones y las combina en una versión de alta resolución.

El método se divide en dos fases: el pre-procesamiento y el de reconstrucción mediante SR.

En la fase de pre-procesamiento intervienen los procesos de: 1) homogeneidad a escala de grises de las imágenes de baja resolución, 2) obtención de una representación espacial común, 3) corrección del contraste, 4) muestreo y 5) mapeado espacial.

Para la fase de SR, se utiliza el *framework* bayesiano para manejar la incertidumbre de los datos localizados en la imagen, proceso con el cual influye la efectividad del proceso de reconstrucción en la imagen con presencia de ruido.

Las imágenes utilizadas se dividen en dos conjuntos:

El primero se compone de cinco imágenes de resonancia cardíaca adquiridos directamente de un escáner 1.5 T de cinco diferentes pacientes. Cada serie contiene entre 2 a 15 plantillas, con una

separación variante entre 8 a 11mm a una resolución 256 x 256. El segundo proviene de un proyecto publico compuesto de 30 imágenes adquiridas de diferentes pacientes, utilizando escáner Génesis Sigma, con una resolución de 256 x 256 pixeles de 10 a 15 plantillas, con una separación variante de 8mm.

Las evaluaciones fueron realizadas mediante las métricas de PSNR y SSIM para comparar el desempeño del método propuesto junto a otro algoritmo de SR. En este caso fue la interpolación bicúbica la que tuvo mejores resultados, los cuales se aprecian en la Tabla 2.1:

Tabla 2.1: Resultados de PSNR y SSIM (Velasco, Rueda, Santa Marta y Romero, 2017)

	Valor de PSNR	
Métrica	Bicúbica	Método propuesto
PSNR	30.39	31.35
SSIM	0.9741	0.9742

Comentarios: se considera que los resultados de este trabajo poco significativos en lo que se refiere a la reconstrucción de imágenes en resonancia. Sin embargo, el uso de técnicas que realcen el aprendizaje estadístico bayesiano aporta una referencia útil para el uso de futuros proyectos similares. Además, la metodología fue clara y el uso de herramientas de código abierto fomenta el uso para software de bajo costo en trabajos académicos.

2.2.3. Técnicas de Súper Resolución para el procesamiento de imágenes médicas

En este trabajo (Isaac y Kulkarni, 2015), los autores presentan técnicas específicas de SR para el procesamiento de imágenes médicas y sus tendencias actuales.

Se presentan conceptos que son fundamentales para comprender el empleo de la SR en el procesamiento de imágenes médicas. Otra mención importante son los tres métodos básicos de SR: compensación del movimiento, interpolación y supresión de ruido/desenfoque.

El tipo de imágenes usadas fueron: Tomografía de rayos X computarizada, Tomografía de emisión de positrones e imágenes de resonancia magnética.

Los resultados muestran que con diferentes tipos de imágenes médicas se presentan las siguientes problemáticas:

- Baja resolución (en los dominios espaciales y de frecuencia).
- Alto nivel de ruido.
- bajo contraste en las imágenes.
- Deformaciones geométricas.
- Presencia de artefactos en las imágenes.

El uso de SR s es de gran importancia para afrontar estos problemas y para extraer información valiosa de las imágenes, la usual es afectada por presencia de ruido e irregularidades debido a la estructura del cuerpo humano y las limitaciones tecnológicas.

También menciona la existencia de una gran cantidad de problemas con respecto a factores como luminiscencia heterogénea y pobre contraste en las imágenes médicas. Para afrontar estos problemas mencionan el uso de técnicas adaptativas de mejoramiento en imágenes, haciendo referencia al algoritmo del PNSD (Projected Normalized Steepest Descent) como una solución adecuada. Adicionalmente, se hace mención de otro algoritmo que es usado para recuperar los detalles perdidos en la frecuencia alta en las imágenes mediante el uso de ventanas; el algoritmo es presentado como SRSW(Super-Resolution by Sparse Weight).

El trabajo en general presenta un reporte de los diferentes tipos de imágenes médicas usados en el procesamiento de imágenes médicas general, explicando que la resolución juega un papel importante en la extracción del conocimiento. También propone que lidiar con la fase del pre-procesamiento de manera dinámica tiene tendencias a mejorar los resultados obtenidos mediante métodos de SR en el procesamiento de imágenes médicas, especialmente si se utiliza en métricas cualitativas de calidad en imágenes.

2.2.4. Reconstrucción regularizada de imágenes de resonancia magnética basada en términos de penalización Hessianos en sistemas CPU/GPU

En este trabajo (Piccialli, Cuomo, y De Michele, 2013), los autores reportan los resultados de la investigación sobre la problemática de la reconstrucción en imágenes de resonancia magnética con pocas muestras.

Cómo principal aportación se tiene un método de reconstrucción basado en derivadas de segundo orden con altos datos de sub-muestreo para obtener una buena precisión en la reconstrucción. También presenta un algoritmo de regularización acelerado, implementado en una unidad de procesamiento gráfico(GPU).

En lo que se refiere a la metodología de la implementación, el algoritmo se presenta de la siguiente forma:

1. Calcular una *máscara(i,j)* iniciando desde el algoritmo de submuestreo.
2. Fija M = (i, j) : máscara $(i, j) = 1$ y (i, j) son las coordenadas del punto $I(i, j)$ en la imagen.
3. Determinar la restricción $C = I \in R^{n^2} : \mathcal{F}(I)_{i,j} = I(i, j), (i, j) \in M$.
4. Obtener la imagen inicial I_0 y procesar $IFFT(I_o)$.
5. **mientras** $k < Iter$ **hacer**.
6. Determinar α_k, β_k, calcular $I_t = I_k - (\alpha_k g(I_k + \beta_k h(I_k)))$ y la proyección de I_t en la restricción fijada, se asume que $\mathcal{F}(It)$ en la transformada de Fourier de I_t y el punto focal es la frecuencia después de la proyección, como se muestra en la ecuación 2.1:

$$PI_t = \begin{cases} K(i,j),\ if(i,j) \in M \\ \mathcal{F}(I_t)_{i,j},\ if(i,j) \notin M \end{cases} \tag{2.1}$$

7. **termina el ciclo**.

Los autores consideran dos cosas antes de iniciar el procesamiento. Una es la regularización que se realiza de manera que se calcule la variación total I y los valores Hessianos H(I) en función de cada uno de los pixeles. Cada pixel es procesado en un hilo dedicado de la GPU. Otra cuestión es el método de proyección, el cual crea una mascara de soporte para la imagen entrante en el dominio de Fourier, antes de empezar la iteración. Este proceso aprovecha todos los núcleos de cada GPU disponible para terminar proceso de reconstrucción.

No hay referencia sobre la base de datos utilizadas o visualización de las imágenes. Pero se menciona que fueron imágenes con resolución de 192 x 220 x 40, 256 x 256 x 256 y 448 x 448 x 488.

El método propuesto es evaluado y comparado con otros dos algoritmos similares, con la métrica PSNR. Los resultados obtenidos se muestran en la Tabla 2.2:

Tabla 2.2: Resultados de PSNR-Piccialli

	Valor de PSNR		
Resolución de la imagen	Método propuesto	IRGNTV	NUFFT
192x220x40	30.23	28.22	25.53
256x256x256	30.64	28.12	26.11

Además, se hizo un análisis de la implementación del algoritmo en CUDA, queriendo evaluar el tiempo de ejecución, siendo medidos en *GFLOPS* y *Speed Up* (el radio entre los *GFLOPS* de un GPU y el CPU) con una GPU 3 x TESLA C1060, con cada núcleo de reloj a 602 MHZ y 240 hilos procesadores, y una computadora Intel core i7 950 a 3.06 GHZ.

Los resultados en *GFLOPS* fueron de 680, 640 y 520 en imágenes de 128 x 128 x 128, 256 x 256 x 256 y 320 x 320 x 320 para el GPU, en el CPU por otro lado fueron de 20, 40 y 50 en imágenes de 128 x 128 x 128, 256 x 256 x 256 y 320 x 320 x 320 respectivamente. Con esto se demuestra la superioridad de procesamiento de las GPU.

Comentarios: el artículo fundamenta la aplicación del cómputo paralelo mediante GPU para procesar algoritmos complejos en tiempo real para imágenes de resonancia magnética.

2.2.5. Una aproximación de Súper Resolución eficiente para obtener imágenes 3D isotrópicas usando múltiples trazos en 2D de MRI

En este trabajo (Hefnawy, 2013), los autores proponen obtener imágenes 3D en alta resolución y contraste mediante un método de SR con parámetros modificados.

La aportación viene siendo la inclusión del optimizador Maximum a posteriori con parámetros de regularización adaptativos locales.

La metodología del trabajo sigue el proceso de utilizar el parámetro regularizado λ para controlar el grado de la reconstrucción en función de la cantidad de ruido que contengan las imágenes. Utilizando un gradiente local para determinar la imagen de alta resolución tomando en cuenta las propiedades de las imágenes médicas. Entonces, usando el Maximum a Posteriori se realiza el proceso de reconstrucción.

La base de datos utilizada se puede observar en el apéndice (A.1) la cual es de libre uso y

contiene varias imágenes del cerebro.

Para presentar los resultados no se mostraron más detalles sobre las características del equipo usado, en cambio, se hizo mención de la resolución de las imágenes de resonancia magnética que es 256 x 256 x 56 voxels y se utilizó las métricas de calidad PSNR como evaluación cuantitativa. Los valores del método propuesto fueron 33.6 dB mientras que otro similar alcanzó 29.5 dB, demostrando superioridad en la reconstrucción. También se mostraron mejoras significativas en lo que se refiere a los bordes.

Comentarios: este trabajo ofrece un panorama adecuado para presentar algoritmos de SR que han sido utilizados generalmente en la literatura, junto con una breve teoría. Ademas, establece que la resolución en imágenes de resonancia magnética es generalmente reducida debido a factores como la intensidad de los gradientes, el ancho de banda, la cantidad de muestreos y las fases de codificación.

2.3. Súper Resolución en otras aplicaciones

En esta sección se presentan los trabajos que se han realizado de SR en diversas áreas del conocimiento.

2.3.1. Una aproximación en conjunto basada en sensado comprimido y Súper Resolución para imágenes de difusión en alta resolución

En este trabajo (Bhavsar y Rajagopalan, 2012), los autores plantean como objetivo obtener imágenes de difusión MRI (dMRI) de alta resolución espacial usando múltiples imágenes de baja resolución con el esquema propuesto de adquisición y reconstrucción de imágenes.

Su principal aportación es combinar los conceptos de censado comprimido y SR para reconstruir datos de difusión en alta resolución. Para ello, proponen combinar técnicas de regularización espacial y de espacio-q para la reconstrucción de datos de alta resolución. Esto se logra dados tres conjuntos de pliegos-gruesos de datos que son subpixeles reducidos cambiados de lugar en función de la dimensión de corte seleccionado. Se termina el proceso de reconstrucción obteniendo una plantilla de dMRI de alta resolución.

El método propuesto utiliza técnicas de regularización espacial y de espacio-q para la reconstrucción de datos de alta resolución, cuya metodología es: dados tres conjuntos de pliegos-gruesos de datos que son subpixeles reducidos cambiados de lugar junto con la dimensión de corte que ha sido seleccionada. Se reconstruye una lámina delgada de dMRI de alta resolución, terminando el proceso de reconstrucción.

La base de datos utilizada fue proveída por la universidad de Florida, referenciada en el apéndice (A.2).

Se realizaron tres experimentos para el método propuesto. En el primero, se generaron de manera artificial imágenes gruesas siendo expuestas a varios tipos de ruido las cuales fueron basadas en el conjunto de imágenes de alta resolución del proyecto *Human Connectome*. El segundo y tercer experimento consistieron en la validación de configuraciones con los datos de un sujeto humano sano. Las evaluaciones fueron hechas utilizando diferentes perspectivas del cerebro y detectando las fibras posibles, a lo cual probó detectar una mayor cantidad.

Comentarios: se observa que el incremento de la resolución de la imagen del cerebro, permitió una visualización de las fibras nerviosas de una forma más clara para los médicos. Esto generara un gran impacto para los estudios de enfermedades que afectan la cognición del cerebro.

2.3.2. Reconstrucción garantizada de Súper Resolución para imágenes

En este trabajo (Graba y cols., s.f.), los autores presentan un nuevo operador de reconstrucción para ser usado en framework's de SR.

La principal aportación es el operador de retropropagación de valores en intervalos que garantiza la reconstrucción de las imágenes que sean sometidas en este método.

La metodología consiste en tres fases:

1. Efectos de translación de muestreo. Primeramente, se obtienen las imágenes de baja resolución mediante el sub-muestreo de la versión de alta resolución usando el modelo de proyección con diferentes valores de traslación.

2. Reconstrucción por una sola entrada. Se genera una señal de la imagen de alta resolución a reconstruir, haciendo una señal imprecisa que contenga los datos complementarios.

3. Reconstrucción por múltiples entradas. Se hace el mismo procedimiento que en la fase 2, con la diferencia de que aquí se grafican varias imágenes de baja resolución para realizar la posterior reconstrucción de la imagen en alta resolución.

El trabajo no hace referencia a ninguna base de datos. Las imágenes fueron simuladas a partir del teorema de proyección, con una resolución de 22610 x 22279.

Los resultados muestran que el método propuesto logró una reconstrucción (en forma de señales) bastante precisa en lo que se refiere hasta el más pequeño detalle. Los autores no mencionan métricas cuantitativas para evaluar la imagen reconstruida.

Comentarios: algo a destacar en este artículo es que propone el uso de imágenes simuladas automáticamente para el proceso de reconstrucción mediante algoritmos que degraden la imagen para extraer la información complementaria en estos.

2.3.3. Mejora de la resolución de imágenes basado en proyección trasera iterativa

En este trabajo (Rasti, Demirel, y Anbarjafari, 2013), los autores presentan un nuevo método de SR para mejorar la resolución de imágenes digitales mediante la iteración de la proyección trasera. Como aportación presenta una nueva técnica que tenga las ventajas de los métodos de interpolación y proyección iterativa trasera para la mejora de la resolución en imágenes.

La metodología del algoritmo se presenta en el siguiente orden:

1. La imagen de baja resolución es interpolada usando diferentes modelos.

2. Se extiende a cuatro imágenes de baja resolución, las cuales son sometidas a filtros de borrosidad.

3. Las cuatro imágenes se usan como datos de entrada para la técnica de proyección iterativa trasera.

4. Se obtiene una imagen de alta resolución, la cual sirve como entrada en el primer paso. El proceso se repite hasta que el valor de la función de error se vuelva menor.

El trabajo no hace mención de ninguna base de datos. Las imágenes que fueron utilizadas son: Lena, Elaine, Pepper y Baboon, con una resolución baja de 128 x 128 y una alta de 256 x 256.

Los resultados, las métricas utilizadas fueron PSNR y SSIM, cuyos valores se presentan en las Tablas 2.3 y 2.4:

Tabla 2.3: Resultados de PSNR (Rasti, Demirel, y Anbarjafari, 2013).

Técnica	**Valor de PSNR**			
	Lenna	Baboon	Peppers	Elaine
Interpolación bicúbica	18.60	19.54	21.07	22.18
Método propuesto con interpolación bicúbica	24.73	20.81	25.91	27.76

Tabla 2.4: Resultados de SSIM (Rasti, Demirel, y Anbarjafari, 2013).

Técnica	Valor de SSIM			
	Lenna	Baboon	Peppers	Elaine
Interpolación bicúbica	0.5827	0.4037	0.6964	0.7186
Método propuesto con interpolación bicúbica	0.8579	0.5827	0. 8604	0.9138

2.3.4. Súper Resolución de múltiples imágenes basada en codificación escasa

En este trabajo (Kato, Hino, y Murata, 2015), los autores presentan un nuevo método de SR basado en codificación escasa para realizar un ajuste correcto de los bloques de sub-pixeles y la utilización de diccionarios hechos para las imágenes de alta y baja resolución. También tiene la característica de que puede operar un número variable de imágenes de baja resolución.

La metodología del trabajo es la siguiente:

1. Se reciben las imágenes de baja resolución y el diccionario de alta resolución.

2. Se selecciona una imagen de baja resolución.

3. Entra en una iteración realizada por cada parche existente

   a) Se realiza un ajuste de los bloques y se seleccionan los parches.

b) Se extraen los pixeles de los bloques seleccionados.

c) Se realiza la codificación escasa.

d) Inicia el proceso de reconstrucción.

4. Se realiza un post-procesamiento mediante proyección trasera.

5. Se repite todo el proceso hasta que se obtenga una convergencia en los pixeles, terminando la reconstrucción.

No hubo una base de datos referenciada. Las imágenes utilizadas fueron del tipo que se usa de manera estándar como Lenna.

Los resultados se presentaron usando la métrica de PSNR, cuyos valores son mostrados en la Tabla 2.5:

Tabla 2.5: Resultados de PSNR (Kato, Hino, y Murata, 2015)

	Valor de PSNR	
Imagen	Interpolación bicúbica	Método propuesto
Lenna	27.91	29.69
Cameraman	27.03	30.19
Flower	35.50	36.61
Girl	31.12	31.98
Parthenon	24.40	25.41

2.3.5. Nueva técnica de interpolación basado en transformadas de Wavelet para la mejora de la resolución en imágenes de satélites

En este trabajo (Rasti, Lüsi, Demirel, Kiefer, y Anbarjafari, 2014), los autores tienen como objetivo mejorar la resolución espacial en imágenes satélites con un nuevo algoritmo de SR basado en intertpolación mediante la transformada estacionaria de Wavelet y la proyección trasera iterativa, dando como aportación la combinación de las ventajas del uso de las transformadas estacionarias de wavelet, la interpolación y la proyección trasera iterativa.

Como metodología se propone los siguientes pasos:

1. La imagen de entrada de baja resolución es interpolada mediante varios modelos.

2. Se descompone a diferentes imágenes de sub-banda mediante las transformadas de wavelet.

3. Cada imagen sub-banda se descompone en cuatro versiones de baja resolución para ser sometidas a efectos de ruido; desenfoque, cambio de escala, rotación y muestreo descendente.

4. Las cuatro imágenes son interpoladas mediante los modelos ya usados.

5. Se utiliza la proyección trasera iterativa para afilar los bordes de las imágenes.

6. La resolución de la imagen resultante es reducida y se regresa al primer paso, de manera iterativa.

El proceso se repite hasta que el valor de la función de error sea mínimo.

Los autores no mencionan el origen de las imágenes satelitales utilizadas. Estás tienen una resolución de 128 x 128 considerado baja y de 256 x 256 para las de alta.

Los resultados fueron evaluados usando las métricas de PSNR y SSIM para comparar el método propuesto junto a otros cuatro algoritmos similares, incluyendo la interpolación bicúbica. Las Tablas 2.6 y 2.7 muestran los resultados:

Tabla 2.6: Resultados de PSNR (Rasti, Lüsi, Demirel, Kiefer, y Anbarjafari, 2014).

	Valor de PSNR			
Técnica	Imagen 1	Imagen 2	Imagen 3	Imagen 4
Interpolación bicúbica	26.44	20.53	21.01	23.74
Método propuesto con interpolación bicúbica	30.28	23.59	24.45	27.63

Tabla 2.7: Resultados de SSIM (Rasti, Lüsi, Demirel, Kiefer, y Anbarjafari, 2014).

	Valor de SSIM			
Técnica	Imagen 1	Imagen 2	Imagen 3	Imagen 4
Interpolación bicúbica	0.6431	0.4505	0.5234	0.5861
Método propuesto con interpolación bicúbica	0.8504	0.7295	0.7824	0.8286

Comentarios: es notable la calidad de presentación de este trabajo. Claro eh informativo sobre el contexto de las técnicas utilizadas. Ademas, hace un especial hincapié a la interpolación bicúbica, lo cual mejora la justificación de su uso en el presente proyecto de tesis.

2.4. CUDA

En esta sección se presentan los trabajos más recientes que se han realizado con la plataforma de cómputo paralelo CUDA como apoyo a la implementación del sistema de procesamiento en un sistema embebido.

2.4.1. Implementación de una imagen integral en GPU

En este trabajo (Chouchene, Sayadi, Atri, y Tourki, 2013), los autores presentan la implementación del algoritmo de imagen integral en una unidad de procesamiento gráfico (GPU) en tiempo real. Dando como aportación la evaluación del rendimiento del modelo de cómputo paralelo mediante CUDA de Nvidia.

Para realizar la evaluación, se ejecutó el algoritmo en imágenes con diferentes resoluciones para ambas unidades (GPU y CPU). La metodología constó de seis pasos fundamentales que fueron:

1. Leer la imagen de entrada.
2. Transformar la imagen en series de datos.
3. Transferir los datos del CPU al GPU.
4. Hacer el cálculo del algoritmo de la imagen integral.
5. Transferir los datos del GPU al CPU.
6. Mostrar los resultados y hacer evaluaciones.

No hubo referencia alguna con respecto a que tipo de imágenes fueron utilizadas.

La implementación fue realizada en Microsoft Visual Studio 2008, CUDA 2.3, con las especificaciones de Hardware Intel Core i5 a 2.6 GHz con 4 GB de RAM y una tarjeta Nvidia GeForce 310M. Se realizaron evaluaciones en imágenes con diferente resolución, tomando como parámetros el tiempo de consumo(TP) en CPU, GPU en milisegundos y el indice de aceleración del *hardware* (Speed up), cuyos valores se presentan en la Tabla 2.8:

Tabla 2.8: Resultados del rendimiento GPU vs CPU (Chouchene, Sayadi, Atri, y Tourki, 2013)

Resolución de imagen	**TP en CPU**	**TP en GPU**	**Speed up**
128*128	0,0099	0,0041	2,41
256*256	0,0129	0,0049	2,63
512*512	0,0178	0,0054	3,29

Es notable mencionar que entre más se incrementa la resolución de la imagen, también el índice de aceleración en el *Hardware*, lo que denota un límite en la resolución de la imagen que la GPU puede manejar. Esto es, con respecto a los límites de cada tarjeta, por supuesto.

Comentarios: este artículo brinda mucha información en lo que se refiere al desarrollo de aplicaciones de visión por computador y procesamiento de imágenes digitales. Además de que aporta resultados que demuestran la superioridad en tiempo de procesamiento de las GPU sobre las CPU convencionales.

2.4.2. Procesamiento de imágenes médicas en GPU: Pasado, presente y futuro

En este trabajo(Eklund, Dufort, Forsberg, y LaConte, 2013), los autores presentan un panorama amplio sobre el procesamiento de imágenes médicas en GPU, incluyendo el cómputo paralelo, historia previa, operaciones básicas, algoritmos y modalidades de las imágenes.

Debido a que la importancia del procesamiento de imágenes médicas en GPU se ha incrementado en los últimos años y ha habido poca atención al respecto, la principal aportación de este trabajo es la de ofrecer a los lectores interesados en el tema una recopilación actual sobre los trabajos de GPU ya hechos en el área clínica.

El trabajo presenta los conceptos de: GPU para el cómputo paralelo, GPU en imágenes médicas, operaciones básicas, algoritmos y modalidades.

GPU para el cómputo paralelo: Las posibilidades tecnológicas del procesamiento paralelo se incrementó bastante gracias a la demanda de la industria de los videojuegos, abriendo camino para darle aprovechamiento en otras aplicaciones como lo es el procesamiento de imágenes.

GPU en imágenes médicas: Gracias al lanzamiento de CUDA en el 2007 por parte de Nvidia, las aplicaciones de imágenes médicas en GPU se volvieron más comunes a la facilidad de manejo que ofrece debido a que ofrece una abstracción a la estructura interna de un GPU. Han sido muchas las modalidades en imágenes por las cuales se ha aprovechado el paralelismo tales como la microscopía, resonancia magnética, ultrasonidos, etc.

Operaciones básicas: En el trabajo se presentaron cuatro métodos de procesamiento de imágenes que han sido usados en GPU, los cuales han sido:

- Filtrado.
- Estimación de histograma.
- Transformación de distancias.
- Interpolación.

Haciendo especial énfasis en los algoritmos de interpolación, donde se es aprovechado al máximo el potencial de las tarjetas gráficas debido a que los procesos de renderizado se sostienen intensamente en cálculos de interpolación, los cuales son extremadamente demandados en la industria de los vídeo juegos, por ende las GPU's han sido diseñadas para dar soporte de *hardware* a la aceleración de estas operaciones, especialmente de imágenes en 3D. Las técnicas de interpolación de alto orden, cómo lo son del tipo cúbico o spline, son procesadas con un mínimo de recursos en las tarjetas gráficas.

Algoritmos: Con respecto a los algoritmos, se presentaron tres tipos en el trabajo, los cuales son: Registro de imágenes, segmentación y supresión de ruido. El primero llega a ser muy común en las implementaciones de GPU, aprovechando su procesamiento rápido en métodos de interpolación para imágenes 3D. La segmentación se usa principalmente en partes del cerebro, vasos sanguíneos, órganos, tumores y los huesos; Tiene tres propósitos generales: comparación, aceleración e interacción visual. Por último está la supresión de ruido, que se ha mantenido en constante uso debido a la naturaleza física de los procesos de adquisición en las imágenes médicas. Los dos algoritmos previamente mencionados son beneficiados por la supresión de ruido en las imágenes de trabajo.

Modalidades de imágenes médicas: Se presentaron nueve modalidades de imágenes, las cuales son:

- Imágenes de tensor de difusión.
- Ultrasonido.
- Imágenes ópticas.
- Microscopía.
- Tomografía computarizada.
- Tomografía por emisión de positrones.
- Tomografía computarizada de emisión monofotónica.
- Resonancia magnética.
- Resonancia magnética funcional.

Según los autores, hay ciertas cosas que se deben tener en cuenta para el trabajo con GPU en imágenes médicas:

1. El ancho de banda entre la GPU y la CPU.
2. Las librerías disponibles.
3. La realización de pruebas y la optimización.
4. La demanda computacional para aplicaciones en tiempo real.

Comentarios: los autores mencionaron que una desventaja de CUDA es que solo soporta tarjetas Nvidia, por lo que mencionan alternativas tales como OpenCL, el cual si tiene soporte para otras tarjetas gráficas como AMD. Otra mención importante fue la herramienta Nvidia Nsight para el proceso de pruebas y depuración, cuyo plugin está disponible en Eclipse IDE desde CUDA 5.0. Esto aporta una ayuda en lo que se refiere al desarrollo de aplicaciones en GPU para usuarios de Linux o MacOS. Detalles como estos deben tenerse en cuenta para la realización de cualquier proyecto que utilice tarjetas gráficas.

2.4.3. Aceleración del algoritmo de detección de bordes basado en la optimización de colonia de hormigas en GPU usando CUDA

En este trabajo (Dawson y Stewart, 2014), los autores presentan una implementación paralela del algoritmo de detección de bordes basado en la optimización de colonia de hormigas en una GPU usando las librerías CUDA de Nvidia. El cual trae como principal aportación ser la primera implementación de su tipo implementado en GPU, teniendo como objetivo mejorar el rendimiento del algoritmo.

Para inicializar el método, la imagen es convertida a escala de grises. Luego, el arreglo de la imagen es cargado en la memoria global del GPU para aplicar una regla proporcional aleatoria que determine la heurística con la cual las hormigas se moverán de un pixel a otro. Las hormigas son representadas en grupos de cuatro por cada hilo de los procesadores en la GPU. Gracias a este algoritmo, las hormigas van encontrando los bordes de la imagen en cuestión.

El entorno de desarrollo constó de : una tarjeta Nvidia GTX y un procesador Intel i7, siendo el código implementado en CUDA toolkit 5.0 para C en el sistema operativo Ubuntu 13.10.

No hubo referencia explícita para la base de datos utilizada, pero se menciona en un apartado que la imagen de Lenna fue la seleccionad para hacer las implementación.

Los resultados se muestran en la Tabla 2.9, tomando en cuenta como factores los hilos por bloque (HB), hilo por hormiga (HHo), hormiga por bloque (HoB) y hormiga por warp (HoW):

Tabla 2.9: Resultados de la colonia de hormigas en GPU (Dawson y S tewart, 2014)

HB	**HHo**	**HoB**	**HoW**
64	62 .679	21 .435	10 .693
128	63 .025	18 .233	6.531
256	62 .858	18 .433	8.884
512	62 .601	18 .431	9.308
768	64 .845	18 .973	9.597

Comentarios: este trabajo provee de una interesante implementación de un algoritmo de detección de bordes basado en un modelo natural para el cual se aprovecha el paralelismo de las GPU en una forma muy concreta mediante la asignación de un número de hormigas por hilos en función de aprovechar eficientemente los recursos de cómputo.

2.4.4. Súper Resolución en tiempo real para transmisiones de vídeo basado en GPU

En este trabajo (Hu, 2014), los autores presentan un nuevo esquema de súper resolución para secuencias de imágenes implementado en un enfoque paralelo basado en GPU.

La aportación de este trabajo es la de presentar una técnica de reconstrucción que estime los patrones de las estructuras existentes en las imágenes naturales para utilizar un proceso de proyección trasera iterativa que produzca los marcos de alta resolución. Otra aportación es la de usar el algoritmo en CUDA de Nvidia para aprovechar el paralelismo en una reconstrucción de imágenes/vídeos en tiempo real.

El algoritmo propuesto consiste en tres partes:

1. **Interpolación inicial en la imagen**: En el módelo de observación (procedimiento crucial para gran parte de los algoritmos de SR) se representa en la ecuación 2.2:

$$y = DBMx + n \tag{2.2}$$

   Donde M, B, D, n son *warping*, *desenfoque*, operadores de sub muestreo y el tratamiento del ruido Gausiano aditivo, respectivamente. y es la imagen de baja resolución modelada y x la versión de alta resolución objetivo.

2. **Estimación del movimiento**:Debido a que es muy difícil tener un buen modelo de movimiento en los algoritmos de SR, en el trabajo se emplea la técnica del descriptor *Local Motion* para describir información con mayor precisión del movimiento.

3. **Fusión y ajuste de los valores perdido en los pixeles**: Se utiliza la técnica de proyección iterativa para minimizar el error de reconstrucción en la imagen de alta resolución objetivo. La ecuación 2.3 muestra el modelo matemático:

$$I_h^{t+1} = I_h^t + G^{BP}(I_l - I_l^t) = I_h^{(t)} + G^{BP}(I_l - GI_h^t) \tag{2.3}$$

   Donde t son las iteraciones, hyl las coordenadas de los pixeles y G^{BP} el operador de proyección trasera.

No se hace referencia alguna a la base de datos, pero las imágenes utilizadas para las pruebas fueron foreman, silent y coastguard, con una resolución de 352x288.

Para los resultados, las pruebas fueron realizadas en el siguiente entorno de desarrollo: Intel Core 2 Duo a 3.00GHz con 3 Gb de memoria, Nvidia GeForce GTX 460. Microsoft Windows XP a 32 bits, Microsoft Visual Studio 2005 y CUDA Toolkit y SDK 4.0. En las Tablas 2.10 y 2.11 se presentan los valores obtenidos del algoritmo propuesto de SR en velocidad de procesamiento y calidad de reconstrucción.

Tabla 2.10: Resultados de la comparación GPU vs CPU (Hu, 2014)

Imagen	**Resolución de entrada**	**Resolución de salida**	**CPU (fps)**	**GPU (fps)**
Foreman	352x288	704x576	4	30.3
Silent	352x288	704x576	4	30.3
Coastguard	352x288	704x576	4	30.3

Tabla 2.11: Comparación entre los algoritmos (Hu, 2014)

Imagen	**Bicúbico (fps)**	**Propuesto (fps)**
Foreman	38.35	34.25
Silent	41.60	27.77
Coastguard	52.36	45.83

Comentarios: este trabajo provee de información valiosa para el lector interesado en algoritmos de SR que trabajen en tiempo real.

2.4.5. Súper resolución mejorada y su implementación en CUDA

En este trabajo (Feng, Zhang, y Gao, 2015), los autores proponen implementar un algoritmo de SR basado en similitud propia en una GPU usando Nvidia CUDA para resolver los problemas de largos tiempos de procesamiento y anomalías en las imágenes

La principal aportación es la de el rediseño del algoritmo de similitud propia para aprovechar la aceleración de la arquitectura en CUDA, añadiendo también un proceso de supresión de ruido.

El algoritmo sigue la metodología siguiente:

1. Se lee la imagen entrante (con una resolución de n x m) desde el archivo binario.

2. Se carga la imagen desde la memoria del CPU a la memoria global del dispositivo GPU (Esto evita ralentizar el proceso debido a la comunicación entre ambos dispositivos).

3. Se procesa la imagen en la GPU. Produciendo la versión de alta resolución.

Se mencionó la base de datos de segmentación Berkeley como referencia del origen de las imágenes aunque no se mencionó un enlace directo. Fueron usadas imágenes con una resolución de 1920x1080 como el principal objeto de estudio.

El entorno de desarrollo constituyó de un CPU Intel Xeon E5620 a 2.40GHz, un GPU GT750Ti Nvidia. Sobre los resultados, se muestran los valores cuantitativos en la Tabla 2.12, donde se evaluó el tiempo de procesamiento que tardó el mismo algoritmo probando tres plataformas: CUDA, C++ y Matlab, siendo implementadas estas últimas dos en CPU.

Tabla 2.12: Resultados del tiempo en procesamiento (Feng, Zhang, y Gao, 2015)

Imágenes (1920x1080)	CUDA	C++	Matlab
1	179ms	53s	347.45s
2	185ms	58s	378.79s
3	164ms	49s	320.62s

Comentarios: este trabajo provee de una justificación en cuanto al uso de las GPU y CUDA sobre otros métodos conocidos. Los resultados mostraron que la implementación superó por mucho a implementaciones hechas en CPU usando el lenguaje de alto nivel C++ y el entorno en Matlab.

2.4.6. Reconocimiento de rotación robusta basado en la coincidencia del contorno usando CUDA en un sistema de automatización

En este trabajo (Sim, Kim, Yuan, Kang, y Cho, 2015), los autores proponen un algoritmo de reconocimiento de objetos para visión por computador en una aplicación para fabricas automatizadas por robots.

Como principal aportación tiene la de utilizar el algoritmo Distancia de Contorno del Centroide para la parte de reconocimiento y hacer su implementación en una GPU utilizando CUDA de Nvidia para el procesamiento en tiempo real.

La metodología se divide en dos partes: Filtrado/etiquetación y el algoritmo de distancia de contorno por centroide.

En el proceso de filtrado suprime el desenfoque de la imagen mediante un filtro Gaussiano como un pre-procesamiento para el etiquetado, con el cual se utiliza el detector de bordes Canny.

Para el proceso de reconocimiento, se propone utilizar un método de análisis de histograma para contrarrestar la rotación y calcular la distancia del centro de gravedad del objeto, sirviendo como base para su reconocimiento.

El entorno de desarrollo constó de lo siguiente: Una computadora Windows 7 de 64 bits con 16GB RAM y un procesador Intel Core i7 de 3.4GHz y como GPU una Nvidia GeForce GTX 560, la resolución de las imágenes fue de 640 x 480.

Los resultados obtenidos en la Tabla 2.13 presentan un análisis del tiempo, tomando como parámetros dos imágenes y el tiempo de procesamiento entre la GPU y CPU en mili segundos:

Tabla 2.13: Análisis de tiempo (Sim, Kim, Yuan, Kang, y Cho, 2015)

Imagen	CUDA	CPU
B	16.18	121.96
C	14.51	117.28

Comentarios: este es un buen artículo para replicar los resultados puesto que menciona las variables utilizadas en sus experimentos. Además, es una referencia importante para aplicaciones en la industria de la automatización utilizando GPU y que muestran una mejora en rendimiento, a comparación de las CPU tradicionales.

2.5. Métricas para evaluaciones cuantitativas

En esta sección se presenta los trabajos que han contribuido a ampliar el conocimiento de las métricas usadas para esta tesis.

2.5.1. Comparación de métodos de interpolación comunmente usados en imágenes

En este trabajo (Han, 2013), los autores presentan una evaluación extensa sobre los posibles algoritmos de mejora de imágenes basados en técnicas de interpolación, usando varios parámetros de medición.

La aportación de este trabajo es hacer un análisis de los algoritmos de mejoramiento en imágenes, presentar sus ventajas y desventajas, realizar comparaciones subjetivas y objetivas mediante experimentos y por último presentar una guía para ayudar al usuario a escoger el algoritmo óptimo, acorde a diferentes aplicaciones.

En las evaluaciones fueron presentados los siguientes algoritmos comunes de interpolación:

- Interpolación del vecino más cercano.
- Interpolación bilineal.
- Interpolación bicúbica.
- Línea B-SP.

De manera en que se pueda realizar la prueba del rendimiento de los algoritmos, se realizaron las implementaciones en MATLAB 2009, utilizando una computadora con las siguientes características: procesador Intel Core 2 T7200 2.00GHz y 2 GB RAM. Para realizar la comparación del rendimiento en los algoritmos se utilizó una imagen de prueba, en la cual se aplicó un acercamiento a la mitad de su sección y se expandió a la resolución original usando un algoritmo diferente.

Se realizaron dos tipos de evaluaciones: subjetiva y objetiva. Para la objetiva se utilizó la métrica de *Signal to Noise Ratio*(SNR) y con la subjetiva se le pidió a 20 personas observar las imágenes mejoradas y dar una evaluación.

No se hizo referencia a ninguna base de datos, sin embargo la imagen de Lenna utilizada en las pruebas.

Para los resultados, se presentan los valores de SNR en la Tabla 2.14, donde fueron utilizadas cinco imágenes para probar los algoritmos del vecino más cercano, bilinear, bicubica y B-SPline están representados como NN, BL, BC y BSP, respectivamente:

Tabla 2.14: Resultados de SNR en 4 algoritmos de interpolación (Han, 2013)

Imagen prueba	**NN**	**BL**	**BC**	**BSP**
1	19.1112	23.3023	23.4204	22.5028
2	16.0059	18.4308	18.5899	17.9319
3	20.0614	25.3736	25.6634	24.3505
4	15.9497	18.7056	18.9392	18.1074
5	17.1252	20.2387	20.9872	18.6582

En cuanto a la evaluación subjetiva, se tomaron en cuenta factores como: sensaciones subjetivas (SS), contorno, veredicto y tiempo de procesamiento (TP). Los valores se muestran en la Tabla 2.15:

Tabla 2.15: Evaluación subjetiva con diferentes métodos de interpolación (Han, 2013)

Interpolación	**SS**	**Contorno**	**Veredicto**	**TP**
NN	Fenómeno de mosaico evidente	No claro	Peor	5
BL	Desenfocado	No claro	Pobre	6
BC	Difuso	Mejora en el fenómeno	Mejor	8
BSP	Relativamente claro	Bordes más claros, sin fenómeno	Bueno	17

Se concluye que la imagen mejorada obtenida mediante interpolación bicúbica ofrece los mejores resultados, seguido por la interpolación bilineal, SPline, y por último la del vecino más cercano.

Comentarios: el algoritmo del vecino más cercano es el más rápido y simple de todos, pero también trae distorsiones significativas entre otros artefactos. La interpolación bilineal es más compleja pero ofrece mejores resultados visualmente, teniendo un balance entre velocidad y calidad visual. La interpolación B-Spline reconstruye de manera suavizada y sin fenómenos extraños, pero el proceso de interpolación llega a suprimir detalles de alta frecuencia causando bordes difusos y trazos falsos. Por último, la interplación bicúbica ofrece la mejor reconstrucción en las imágenes, pero es computacionalmente más costosa.

2.5.2. Criterio perceptivo para evaluaciones sobre la calidad en imágenes

En este trabajo (Pappas, Safranek, y Hill, 1999), los autores propusieron examinar los criterios de evaluación que se toman en cuenta para medir la calidad en las imágenes.

La principal aportación es la de brindar los fundamentos teóricos sobre las métricas de evaluación existentes y su uso en las imágenes.

Se presentaros varías métricas que son basadas en el sistema visual humano, considerando tanto las de propósito general como las específicas para cada aplicación. Sin embargo, existen parámetros que deben tomarse en cuenta para evaluar de manera perceptiva una imagen. Estos son:

- Imágenes de referencia/objetivo.
- Registro.
- Calibración.
- Modelo de despliegue.

La imagen de referencia es el modelo que se tiene para hacer la comprobación de si una imagen es buena o mala; esta tiene que ser perfecta o incorrupta. La calibración se requiere para realizar la normalización de datos debido a que la imagen puede venir desde distintos dispositivos de adquisición. El registro es la correspondencia punto a punto entre dos imágenes, haciéndolas homogéneas. Por último, el modelo de despliegue es el medio por el cual se representa la imagen. Estos parámetros son usados para el desarrollo de nuevas métricas.

Existen también aquellos modelos generales que forman métricas basadas en el procesamiento a bajo nivel del sistema visual humano, son de propósito general y algunas vienen siendo:

- **Métricas basadas en el análisis de frecuencia**: Comprende de una jerarquía de filtros que descomponen la imagen en varios componentes o canales con diferentes frecuencias espaciales.
- **Métricas basadas en sensibilidad de contraste**: La naturaleza de éstas métricas comprende lo que es la determinación del monto energético en cada componente que es requerido para localizar la imagen objetivo a escala de grises.
- **Métricas basadas en enmascaramiento de luminancia**: Trabajan sobre la normalización de los valores en la imagen para hacer las conversiones propias en cuestión de luminancia.

Los modelos que se definen por el color adoptan la descomposición de frecuencias dado un codificador en específico. Son muy efectivas en lo que se refiere al rendimiento de procesamiento. Algunas métricas que han surgido a partir de estos modelos son:

- **La métrica Safranek-Johnston**: Usa un modelo de enmascaramiento empírico el cual utiliza para determinar la sensibilidad de ruido que existe en cada componente dada una imagen a escala de grises.
- **Métrica Watson DCT**: Usando la transformada discreta del coseno, está métrica procesa la visibilidad de los umbrales para los coeficientes de la transformada, proveyendo una métrica sensible a los codificadores basados en bloques.

Existen otros modelos de los cuales surgieron basándose en las características de los antes mencionados, muchos de ellos ya desarrollados para una aplicación en específico. Las métricas para vídeos comparten la mayoría de las características que para imágenes, con la diferencia de que tienen otros parámetros como frecuencias espaciales, temporales, etc.

Comentarios: el trabajo presenta varías métricas de evaluación que extienden el repertorio de evaluaciones en imágenes digitales. También presentan ciertos conceptos interesantes como lo es la fidelidad de la imagen, que viene siendo el principal foco de interés al momento de evaluarlas en función de que tan aproximada es para representar la realidad que captura. Otro concepto es que las métricas de propósito general son más complejas y costosas de aplicar, por el contrario de las específicas que vienen siendo aún más simples.

2.5.3. La evaluación de calidad en imágenes: Desde la Visibilidad del Error a la similitud estructural

En este trabajo (Wang, Bovik, Sheikh, y Simmoncelli, 2004), los autores proponen un *framework* alternativo para la evaluación de calidad en las imágenes basado en la degradación de la información estructural.

La aportación de este trabajo es el uso de la evaluación de imágenes basándose en el error por sensibilidad.

La metodología del *framework* usando la evaluación *Basado en Sensibilidad de Error* y la *similitud estructural*, es presentada. Utilizando una imagen de referencia, el primer método consta de cinco fases:

1. **Pre-procesamiento**: Se realizan una variedad de operaciones básicas para eliminar las distorsiones conocidas de las imágenes a comparar.

2. **Filtrado FSC (Función de sensibilidad en contraste)**: Describe la sensibilidad del sistema visual humano a diferentes frecuencias temporales y espaciales que están presentes en los estímulos visuales.

3. **Canal de descompisición**: Se separan las imágenes en varios canales mediante una selección en criterio de frecuencia temporal, espacial y la orientación.

4. **Normalización del error**: Se calcula la diferencia de error entre la referencia descompuesta y la señal distorsionada para realizar una normalización, acorde al modelo.

5. **Agrupación de errores**: Combinando todas las señales de error normalizadas sobre la extensión espacial de la imagen y sus diferentes canales en un solo valor, se termina el proceso de evaluación.

Ahora, para la similitud estructural, son cuatro fases que vienen siendo:

1. **Medida de luminancia**: De forma paralela se mide la luminancia de la imagen de referencia y la evaluada.

2. **Medida de contraste**: De la misma forma paralela, se mide el contraste entre la imagen de referencia y la evaluada.

3. **Comparación entre el contraste, luminancia y estructura**: Los valores obtenidos anteriormente se comparan junto con el valor estructural.

4. **Combinación de todos los valores**: Por último, se combinan las comparaciones para obtener la medida de similitud entre las dos imágenes.

Terminado este proceso, se obtiene la medida de similitud estructural para la evaluación de la imagen utilizada.

La base de datos utilizada consta de imágenes de 24 bits con una resolución promedio de 768 x 512 con la modalidad JPEG y JPEG2000. El enlace se encuentra en el Apéndice (A.3).

Para los resultados, se midió el rendimiento de los algoritmos de manera subjetiva (la cual es foco de este trabajo) la cual quedó expuesta en las imágenes evaluadas y requiere de ser juzgadas por el lector. Según los autores el método propuesto obtiene mejores resultados que los otros modelos comparados.

Comentarios: en este trabajo se presentan conceptos que se deben tomar en cuenta al momento de realizar evaluaciones cualitativas. Uno es que las imágenes digitales están expuestas a varias distorsiones durante los procesos de adquisición, procesamiento, compresión, almacenamiento, transmisión y resolución, provocando una degradación en la calidad visual, a lo cual se debe tener presente las problemáticas de su adecuada adquisición.

También se expuso que cuando las evaluaciones son hechas por humanos, la subjetividad es el único método correcto. Sin embargo, para ciertos casos es mejor la objetiva debido a que es menos costoso tanto en tiempo como en dinero. Las ventajas de usar métricas objetivas son: monitorear y ajustar la calidad de las imágenes y optimizar los algoritmos en sistemas de procesamiento en imágenes.

2.5.4. Eliminación de ruido impulsivo en imágenes a color, utilizando interpolación con funciones de base radial

En este trabajo (Alberto, 2011), los autores proponen suprimir el ruido impulsivo en imágenes a color mediante un método de interpolación a través de funciones de base radial.

La principal aportación es que este método, a diferencia de otros algoritmos de supresión de ruido, solo altera el valor de los pixeles con presencia de ruido mediante un criterio definido.

La metodología del trabajo se comprende a continuación:

1. Revisa los canales R, G y B uno a uno.
2. Revisa cada pixel en el respectivo canal.
3. Si es un pixel ruidoso.
   a) Filtrar el vecindario del pixel ruidoso.
   b) Interpolar el pixel ruidoso en función de su vecino filtrado.
   c) Se termina el ciclo de pixeles.
   d) Se termina el ciclo de los canales.
   e) Unificar los canales.
4. Se termina el ciclo de pixeles.
5. Se termina el ciclo de los canales.
6. Unificar los canales

El trabajo no hace referencia a una base de datos en específico, pero utilizaron la imagen de *Lenna* con una resolución de 512 x 512 pixeles, como medio para sus pruebas.

Para la obtención de los resultados se expuso el entorno de desarrollo: Una computadora con un CPU de doble núcleo a 3.2 GHz, 2 GB RAM, con las implementaciones hechas en MATLAB 7.1.

Para la evaluación cuantitativa se utilizaron las métricas del error cuadrado medio (ECM) y *Peak Signal to Noise Ratio* (PSNR), dando como resultado el valor 14 para ECM y 35.8d db en PSNR con 40 % de ruido en la imagen.

Comentarios: este artículo presenta un buen fundamento de que la métrica PSNR y ECM muestran evaluaciones válidas para la reconstrucción en imágenes.

2.5.5. Análisis de la calidad diagnóstica de imágenes de tomografía computarizada procesadas con un filtro bilateral

En este trabajo (Adame y Clemente, 2012),los autores proponen evaluar la calidad de diagnóstico en imágenes de tomografía computarizada que son procesadas mediante un filtro bilateral.

Tiene como principal aportación la evaluación del filtro bilateral trabajando con ruido estadístico simulado.

La metodología de los experimentos consta de realizar el análisis de imágenes mediante las herramientas ROC(Receiver Operating Characteristic) y LROC (Localization ROC) en comparación con tres especialistas con experiencia de más de cinco años. Las regiones de interés fueron seleccionadas a partir de los siguientes criterios:

- **Área**: El número real de píxeles en la región.
- **Orientación**: Ángulo que forma el eje X con el eje máximo de la elipse que contiene a la región.
- **Centroide**: Centro de masa de la región.

Luego, determinando una zona correcta de localización, se llevó a cabo la evaluación entre los observadores humanos y los algoritmos.

La base de datos se obtuvo a través de un equipo de tomografía, generando 100 imágenes con presencia de accidentes cerebro-vasculares con una resolución de 512 x 512.

Los resultados fueron obtenidos utilizando las métricas de similitud estructural (SSIM), similitud estructural basado en el borde (ESSIM), diferencia media absoluta (MAD) y la relación señal a ruido de pico(PSNR). Los valores obtenidos se presentan en la Tabla 2.16:

Tabla 2.16: Métricas objetivas para las imágenes con y sin ruido (Adame y Clemente, 2012)

Métrica	**Imágenes con ruido**	**Imágenes con filtro bilateral**
SSIM	0,7288	0,9998
ESSIM	0,9963	0,9992
MAD	6,4424	0,0368
RSR	7,36893	14,6784

Donde se mostró un aumento en los valores de las métricas a excepción en la MAD, representando un incremento en la calidad de las imágenes al ser filtradas de ruido.

Los observadores humanos evaluaron las imágenes antes y después del ruido, declamando que se obtuvo una mejora considerable en la localización de las áreas de interés.

Comentarios: este trabajo tiene ciertas nociones valiosas en lo que se refiere a cómo un sistema de procesamiento de imágenes debe mejorar el diagnóstico realizado por los médicos expertos. También hay que tomar en cuenta que la tomografía computarizada llega a ser benigna debido a que usa radiación ionizante, por lo tanto se deben encontrar soluciones a los largos lapsos de adquisición de una imagen por tomografía. Una mención especial de interés es que los autores usaron el software *Image J* para procesar las imágenes.

2.6. Sistemas embebidos

En esta sección se provee de los trabajos realizados en el estado del arte sobre sistemas embebidos, exponiendo la fundamentación de su uso en el aspecto académico e industrial.

2.6.1. Construcción de una plataforma de cómputo móvil con NVIDIA Jetson TK1

En este trabajo (Su y cols., 2015), los autores proponen mostrar las posibilidades para nuevas direcciones de investigación que tiene la tarjeta de desarrollo NVIDIA Jetson TK1, mediante la construcción de una plataforma de cómputo móvil.

Como aportación, brinda mostrar cuales son las ventajas de usar una GPU embebida sobre las CPU de escritorio en cuestión de precio, consumo energético y movilidad. Propone argumentar esto con la implementación de dos herramientas creadas especialmente para la plataforma. Los autores esperan mostrar una alternativa más versátil para implementaciones en GPU.

La metodología del trabajo consta de cuatro fases:

1. **Configuración del sistema**: Para comenzar, se necesita realizar ciertos parámetros para iniciar la plataforma tales como la actualización del sistema de imágenes, configuración de la red e instalar ciertos paquetes de Software.

2. **Configuración de puertos**: Se necesita compilar ciertos comandos y códigos de CUDA ClustalW para utilizar las herramientas propuestas. También, se requiere regular y modificar la librería CUDA-MCC.

3. **Sincronizar el rendimiento**: Es necesario hacer una sincronización entre la Jetson TK1 y el GPU de escritorio debido a la diferencia de cantidad entre los bloques de hilos.

4. **Diseño de la interfaz**:Para este parte se realizaron dos interfaces gráficas mediante Qt para las herramientas propuestas.

El entorno de desarrollo constó de: Computadora de escritorio con Intel Xeon CPU de 2.0 GHz, cuatro tarjetas gráficas Nvidia Tesla K20m con 4.8GB RAM. La tarjeta Nvidia Jetson TK1 tiene un CPU ARM, con una GPU Tegra K1 y 2GB RAM.

Dadas las observaciones de la implementación, se concluyó que:

- Ambas plataformas lograron un índice de aceleración por tres.
- La relación costo/rendimiento fue mejor en la Jetson TK1.

Comentarios: este artículo provee de información literaria en lo que se refiere al uso de sistemas embebidos usando GPU, la cual actualmente es escasa.

2.6.2. Estimación estéreo embebida en tiempo real mediante coincidencia semi-global en GPU

En este trabajo (Chac y Espinosa, 2016), los autores proponen implementar y evaluar un sistema de visión en tiempo real en los nuevos dispositivos GPU embebidos.

Como principal aportación se tiene la de implementar un método de reconocimiento mediante estimación por disparidad en la tarjeta embebida Jetson TX1 y así demostrar la viabilidad de utilizar sistemas embebidos en tiempo real, tomando como principales factores el consumo energético, rendimiento y movilidad.

Para la implementación, se utilizó la siguiente metodología para hacer la estimación por disparidad, el cual consta de cinco fases:

1. Las imágenes son copiadas de la memoria del huésped a la del GPU.
2. Se extraen las características de cada imagen y se usan para una comparación por similitud para generar un costo de coincidencia local por cada pixel y su posible disparidad.
3. Se agrega un costo de suavizado para reducir el índice de error.
4. Se procesa la disparidad y es aplicado un filtro 3x3 de la media para remover *outliers*.
5. La disparidad resultante de la imagen es copiada a la memoria del huésped.

Se utilizó el siguiente entorno de desarrollo: Una tarjeta Nvidia Tegra x1 con 8 núcleos ARM y un consumo energético de 10W, una tarjeta Nvidia Titan X con un consumo de 250W, y para los experimentes se utilizaron sólo imágenes con resolución de 640x480 pixeles.

De los resultados, se hizo un comparación en función del rendimiento. La tarjeta Titan es 10 veces más potente que la X1 (como se esperaba, debido a la diferencia de características entre los dispositivos), pero la X1 tuvo mejor rendimiento en Wats, tomando en cuenta que procesó imágenes a 42 marcos por segundo con gran precisión en tiempo real. Con esto se demostró que los GPU en sistemas embebidos de bajo consumo son plataformas sobresalientes para el procesamiento de vídeos/imágenes en tiempo real.

Comentarios, este trabajo aporta la importancia de los sistemas embebido en el uso de aplicaciones que demandan gran capacidad de procesamiento. En especial los de visión por computador. Con la adición de las GPU es posible aprovechar las ventajas de la movilidad y bajo consumo de un sistema embebido junto con el procesamiento paralelo.

2.6.3. Detección de peatones basada en GPU para conducción autónoma

En este trabajo (Campmany y cols., 2016), los autores proponen implementar un sistema de detección de peatones en tiempo real para el sistema embebido Nvidia Tegra X1.

Se presentan como aportación la implementación de un sistema en detección de peatones que alcanza los requerimientos necesarios para el procesamiento en tiempo real que demanda la conducción de automóviles autónomos.

La metodología del proyecto utilizó los algoritmos de extracción de características siguientes: histograma de patrones binarios locales, histograma de gradientes orientados y ventana deslizante piramidal junto con máquina de vector soporte. La implementación se divide en cinco fases:

1. Se copia las imágenes de la memoria del CPU huésped a el dispositivo GPU.
2. Se crea una pirámide a escala de las imágenes.
3. Se extraen las características de cada plantilla de la pirámide.
4. Se hace la segmentación y clasificación de las ventanas por cada plantilla.
5. Se copian los resultados de la detección a la memoria del huésped para hacerles un proceso de refinación usando Supresión No-máxima.

Se utilizó el siguiente entorno de desarrollo: Una computadora con Intel i7, dos tarjetas gráficas; una Nvidia GTZ 960 y una Tegra X1. Como factor de evaluación se toma que: se considera tiempo real el procesamiento de marcos por segundo con valores entre 20 y 40.

Los resultados mostraron que la a GPU logró un procesamiento de 263 marcos por segundo, con la Tegra X1 teniendo 40 y la CPU apenas llegando a 4. Mientras que el desempeño de la tarjeta para escritorio fue muy superior, la versión embebida cumple las expectativas necesarias para el procesamiento en tiempo real y dejando atrás los valores obtenidos por el CPU de manera significativa.

Comentarios: este trabajo es valioso para futuras aplicaciones de Inteligencia Artificial con respecto a los autos autónomos, la cual es una línea de investigación muy interesante y lucrativa. Provee de referencias a otros trabajos de detección automática en GPU, cuyos proyectos son similares en la aplicación de algoritmos. Además, en los resultados de este trabajo se probó la superioridad de procesamiento que tiene la Tegra X1 con respecto a un CPU i7 de intel, los cuales son considerados como la serie de procesadores más potente del segmento.

2.6.4. Un sistema embebido para la detección de personas en tiempo real en imágenes/vídeos infrarrojos usando Súper Resolución y técnicas de características de Haar

En este trabajo (Ramos y Garcia, 2015), los autores proponen integrar los algoritmos de detección de personas y de súper resolución para el reconocimiento de estas, siendo implementado en un sistema embebido.

Dando como aportación los siguientes tópicos:

- La implementación de un mejor algoritmo de detección en condiciones nocturnas usando una cámara infrarroja.
- Integrar la súper resolución en el sistema de detección de personas propuesto.
- Implementación del método diseñado en un sistema embebido para aplicaciones de tiempo real.

La metodología se divide en tres pasos fundamentales:

1. **Súper Resolución**: En este paso, se implementan los diferentes algoritmos clásicos de SR en imágenes/vídeos de baja resolución.
2. **Detección de personas usando las características de Haar**: Aquí se analizan los métodos de Haar para el proceso de detección.
3. **Variación de las características de Haar**: Se implementa una variación en las características de Haar con el propósito de mejorar la velocidad de procesamiento
4. Implementación del algoritmo de detección de personas con SR en un Raspberry pi 2.

Las imágenes fueron creadas por los autores.

Los resultados fueron obtenidos utilizando las métricas de SSIM y PSNR, donde se evaluaron los tres algoritmos de SR mediante interpolación bicúbica, bilineal y del vecino más cercano. Para PNSR los valores fueron 30.08, 29.43, 28.40 y para SSIM dieron 0.8892, 0.8878 y 0.8612, respectivamente. La interpolación bicúbica fue la mejor de las técnicas implementadas.

2.6.5. Procesamiento de imágenes en tiempo real implementado en computadoras embebidas de bajo costo

En este trabajo (Shah, 2014), los autores propusieron construir, evaluar y mejorar el rendimiento de un sistema de visión implementado en una computadora embebida, siendo esta incrustada en un drone.

La principal aportación de este trabajo es demostrar la valides de usar sistemas de visión artificial, en vehículos no tripulados, usando computadoras embebidas de bajo costo.

La metodología de funcionamiento consta estados de máquina, los cuales son:

1. FLYING.
2. SEEK HOME.
3. LAND HIGH.
4. LAND LOW.
5. POWER OFF.

El estado FLYING se inicia cuando el drone está ya en vuelo. En el momento que la estación de tierra da una orden de aterrizaje, pasa al estado SEEK HOME. Aquí se usa el piloto automático para regresar al lugar inicial de lanzamiento. De aquí transita al estado LAND HIGH, donde usa los algoritmos de visión para guiar al drone a la zona de aterrizaje. Llegada a una cierta altitud se transita al estado LAND LOW, donde se regula de manera específica la velocidad de los motores para hacer un descenso seguro. La máquina de estado termina cuando en la lectura de los sensores a presión le indica al sistema que el drone a llegado a tierra, pasando al estado POWER OFF y apaga el equipo.

Para los resultados se utilizó el siguiente entorno de desarrollo: Plataformas BeagleBone Black, HardKernel Odroid XU, una camara Logitech C920, sistema operativo Ubuntu Linux ROS Hydro y la librería OpenCV. La precisión de aterrizaje fue de 195.33 cm alejado de la zona planeada. Habiendo realizado las pruebas en el interior del laboratorio y con la cámara colocada en el centro del drone.

Comentarios: este artículo presenta el fundamento de que las aplicaciones en visión artificial y robótica son posibles cuando se implementan en sistemas embebidos, en especial con factores como el tiempo de desarrollo y movilidad. También habla sobre artículos de drones que se han extendido para su uso en la agricultura, eso junto con una predicción de 82 billones de dolares en actividad económica.

2.7. Análisis del estado del arte

Muchos de los trabajos propuestos a lo largo del estado del arte en los últimos años proveen un fundamento intelectual en el tema de SR. En el área médica se han visto trabajos que presentan nuevos algoritmos de SR para la mejora de imágenes tales como las del tipo MRI. Dichas mejoras incluyen: el manejo de escasez de datos, supresión de ruido, procesamiento en datos de sub-muestreo, inclusión de nuevos operadores para optimización, etc. Estas aportaciones han contribuido a la reconstrucción y mejoramiento de MRI.

Otros trabajos han propuesto técnicas similares combinando diferentes modalidades de imágenes para obtener mejoras en la resolución. Algunos proponen el uso de imágenes médicas simuladas para replicar ciertas condiciones de campo sin requerir buscar en alguna base de datos. En las aportaciones se incluyen nuevos métodos de SR combinando diversas técnicas clásicas con otros paradigmas como lo son las redes neuronales.

La SR también se ha extendido en las áreas de *machine learning* a lo cual se han presentado nuevos conceptos como lo son los "diccionarios": bases de datos propias de los métodos de una sola imagen que tienen la función de entrenar los algoritmos de aprendizaje en reconstrucción.

Existen trabajos de alto impacto que mejoran imágenes satelitales que combinan varias técnicas de procesamiento, incluyendo la interpolación, para tratar con imágenes de una resolución a gran escala.

Sobre los trabajos realizados en CUDA, un objetivo de interés común es comparar las ventajas del uso de GPU's sobre CPU's convencionales para diversas aplicaciones (como el procesamiento de imágenes). Muchas citas en los trabajos testifican que las GPU están especialmente diseñadas para trabajar algoritmos de interpolación, justificando aún más su uso para este proyecto de tesis.

Los trabajos revisados sobre las métricas cuantitativas ofrecen expandir el panorama sobre las futuras aportaciones en medición de imágenes. Nuevos parámetros fueron presentados y evaluados mediante experimentación para la creación de guías que instruyan en su apropiado uso de las métricas.

Por último, los trabajos en sistemas embebidos muestran las claras ventajas que se obtienen al usarlos en aplicaciones de alto rendimiento. La tarjeta Jetson TK1 fue presentada como alternativa para la construcción de una plataforma de cómputo móvil. Un segmento más avanzado de la Jetson fue usado en la detección automática de peatones en automóviles autónomos. Otras tarjetas de desarrollo similares han servido como base de implementación para aplicaciones de visión por

computadora, demostrando su viabilidad en pruebas de campo.

Con ya un establecido antecedente, el impacto de la SR se ha incrementado considerablemente en la última década gracias a la demanda de imágenes con resolución más alta en el área médica.

Capítulo 3

Algoritmos de Súper Resolución y métricas de evaluación

En este capítulo se presentan los algoritmos de súper resolución seleccionados. Se explica el funcionamiento y complejidad de cada uno de los métodos que son: el vecino más cercano, interpolación bilineal y bicúbica. También se describen las métricas de evaluación utilizadas para comparar los algoritmos mencionados, considerando los criterios de calidad de imagen y tiempo de procesamiento.

3.1. Introducción

Súper resolución son los métodos de procesamiento en señales que construyen imágenes de alta resolución a partir de la información complementaria, que extraen de una o varias imágenes de baja resolución provenientes de la misma escena (Yue, 2016).

3.1.1. La importancia de la súper resolución

Existen varias limitaciones técnicas en la adquisición de imágenes digitales (Salvador, 2017), las principales son:

- Presencia de distorsiones ópticas y desenfoque de los lentes, expresado típicamente como la función de dispersión de punto (PSf-Point Spread Function).
- Muestreo insuficiente en cuestión de densidad y alineamiento en los sensores.
- Desenfoque por movimiento debido a la baja velocidad del obturador.
- Presencia de ruido debido a las limitaciones del sensor y algoritmos de compresión con perdida, siendo el resultado de transmisiones y almacenamientos coaccionados.

La manera en que la SR quedaría obsoleta es sí se ignora el balance entre las capacidades técnicas y soluciones económicas e intentar construir dispositivos con una resolución espacial (o temporal) excelente, a un precio de un producto excesivamente costoso para el mercado. Algunas soluciones relacionadas al hardware que intentan lograr esto son:

- Reducir el tamaño de pixel a coste de más ruido visual.
- Incrementar el número de sensores de pixeles, aumentando la capacitancia.
- Reducir la velocidad del obturador, incrementando el ruido.
- Adoptar técnicas de óptica y sensores más precisos, incrementando el precio del dispositivo.

  La solución alternativa, proveída por la SR, es la de usar herramientas de software de manera que se obtenga la salida de alta resolución. La ventaja del post procesamiento de los datos visuales capturados es que permite intercambiar el costo de software por el de hardware (las soluciones por software son generalmente más económicas), lo cual resulta en un producto final que tendrá un precio más bajo en el mercado y que será compatible con dispositivos y sistemas de imágenes contemporáneos.

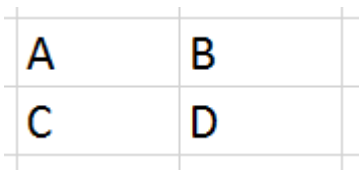
3.1.2. Concepto de súper resolución

Principio de amplificación en imágenes

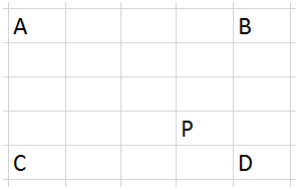
El aumento en imágenes se define como un proceso de conversión de una imagen de baja resolución a su versión de alta resolución. Actualmente, el método común para el aumento en imágenes es la interpolación (Han, 2013). El principio básico del aumento en imágenes es el de incrementar el número de pixeles en un conjunto de imágenes de baja resolución, siendo este el proceso de conversión a una versión de alta resolución.

Para comprender mejor el concepto, se presenta un ejemplo simple y breve:

Para una imagen de resolución n x m, tomando como ejemplo la Figura 3.1, el valor original de intensidad en los cuatro pixeles marcados como A, B, C, y D en (a) fueron trasladados en nuevos lugares mostrados en (b), acorde al factor de aumento. Pero, debido a la extensión espacial de la imagen, grandes números de pixeles entre A, B, C, y D han aparecido, cuyos valores son desconocidos, como lo es P. Por lo tanto, los valores de esos pixeles deben ser estimados mediante interpolación.



(a) Imagen sin amplificar.



(b) Acercamiento de la zona.

Figura 3.1: Principio de amplificación de imágenes.

3.2. La complejidad de los algoritmos

La complejidad de los algoritmos hace referencia en las ideas de como los algoritmos programados en computadoras son procesados. Medir en tiempo no es realmente objetivo debido a la heterogeneidad de los entornos de implementación. Por lo que, la manera ideal de medir la complejidad es

mediante el comportamiento asintomático (Zindros, s.f.). Una manera de entender este análisis es como un "filtro"que remueve los factores heterogéneos de un algoritmo y solo mantienen su término creciente más grande. Todo representando en forma de una función. Haciendo un ejemplo:

El comportamiento asintomático de un algoritmo representado en la siguiente función 3.1:

$$f(n) = 2n + 8 \tag{3.1}$$

donde n son el número de instrucciones en el programa, de manera asintomática está descrito en la ecuación 3.2:

$$f(n) = n \tag{3.2}$$

Usando a θ describimos que el algoritmo tiene una complejidad descrita en la ecuación 3.3 :

$$f(n) = n \rightarrow \theta(n) \tag{3.3}$$

3.3. Algoritmos de súper resolución seleccionados

A continuación se presentan los tres algoritmos clásicos de súper resolución seleccionados para este trabajo.

3.3.1. Vecino más cercano

Comenzando con la interpolación del vecino más cercano (Han, 2013), su método está descrito de la siguiente manera: dado la posición de un pixel P en la imagen resultante, junto con el valor de las distancias de P entre sus valores vecinos A, B, CyD, los valores de color (para cada una de las bandas) del pixel P se fijan de acuerdo al pixel que se encuentre más cerca de P. Se muestra un diagrama de flujo del algoritmo en la Figura 3.2).

Presentando un ejemplo en la Figura 3.3 , se asumen los siguientes 5 puntos en una imagen; $(i, j), (i, j + 1), (i + 1, j), (i + 1, j + 1)$ y (u, v). Los primeros cuatro puntos son los vecinos del quinto que es el valor desconocido P a encontrar. Al calcular la distancia entre ellos se fija al valor desconocido P el valor del punto más cercano.

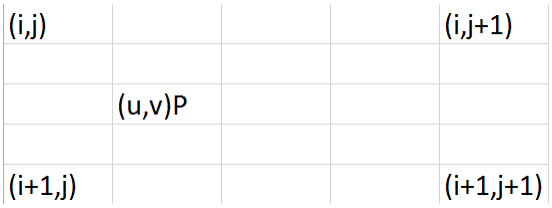


Figura 3.3: Diagrama del vecino más cercano.

La complejidad del algoritmo se expresa en la ecuación 3.4

$$\theta(n^2) \tag{3.4}$$

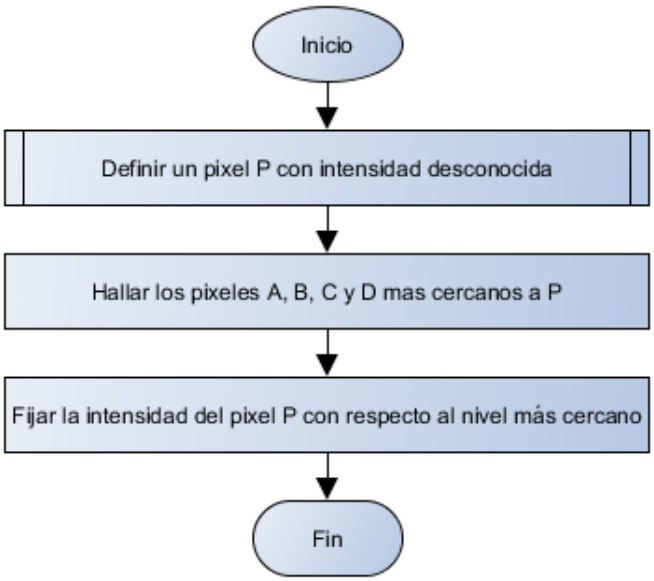


Figura 3.2: Diagrama de flujo del vecino más cercano.

que está en función de las características de la imagen.

3.3.2. Interpolación bilineal

Siguiendo de manera ascendente, en cuestión de complejidad, viene la interpolación bilineal (Han, 2013). El principio es similar al vecino más cercano. Dada la posición de un pixel desconocido P de la imagen resultante, este se reúbica a la imagen original, entonces se utiliza la influencia de **los cuatro pixeles A, B, C y D** para estimar los valores desconocidos de P. En el valor de intensidad de P tendrán mayor influencia los pixeles más cercanos a este. Se tiene una mejor apreciación en la Figura 3.4.

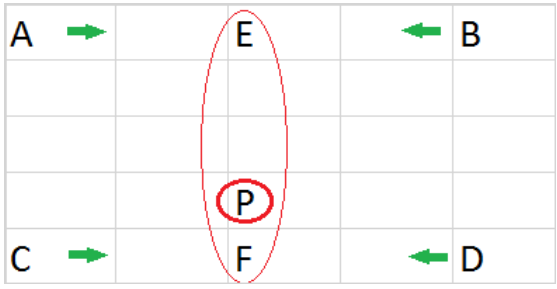


Figura 3.4: Diagrama de la interpolación bilinear.

Ahora, para presentar un poco más a detalle de manera analítica, se asume el siguiente ejemplo:

Las coordenadas A, B, C y D son $(i, j), (i, j + 1), (i + 1, j)$ y $(i + 1, j + 1)$ y P es (u, v). La interpolación bilineal consiste en los siguientes tres pasos:

1. Estimar la influencia de A, B y denotarla como E, mostrado en la ecuación 3.1.

$$f(i, j+v) = [f(i, j+1) - f(i, j)]v + f(i, j) \tag{3.5}$$

2. Se estima la influencia de C y D y se denota como F, mostrado en la ecuación 3.2.

$$f(i+1, j+v) = f(i+1, j+1) - f(i+1, j)]v + f(i+1, j) \tag{3.6}$$

3. Se estima la influencia de E y F y se denota como P, expresado en la ecuación 3.3.

$$\begin{aligned} f(i+u, j+v) = (i-u)(1-v)f(i, j) - (1-u)vf(i, j+1) + \\ u(1-v)f(i+1, j) + uvf(i+1, j+1) \end{aligned} \tag{3.7}$$

El algoritmo es presentado en un diagrama de flujo en la Figura 3.5.

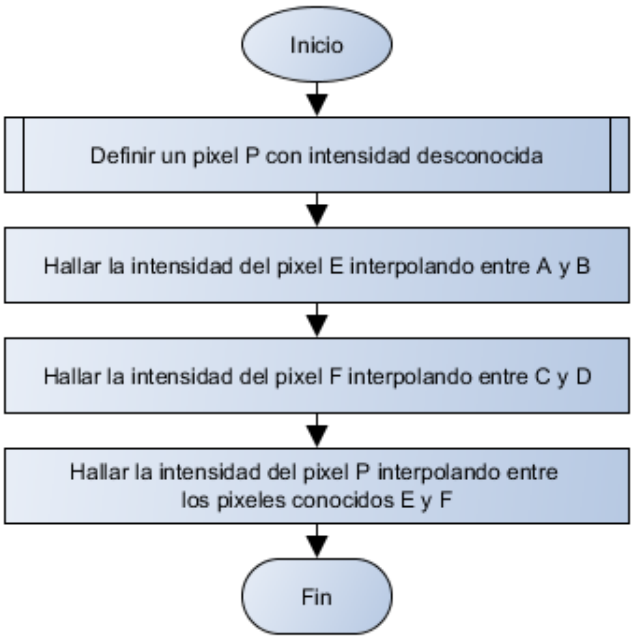


Figura 3.5: Diagrama de flujo de la interpolación bilineal.

La complejidad del algoritmo se expresa en la ecuación 3.8.

$$\theta(n^2) \tag{3.8}$$

que está en función de las características de la imagen. Su similitud con el vecino más cercano es debido a la cantidad de procesos que debe hacer y no por la complejidad matemática en si.

3.3.3. Interpolación bicúbica

Por último, presentando el algoritmo más extenso, la interpolación bicúbica (Han, 2013) es similar a la bilineal; para encontrar el valor de un pixel P en una imagen en la imagen resultante, el rango de influencia se extiende a 16 pixeles adyacentes, por lo que el valor del color del pixel P es estimado mediante los 16 pixeles vecinos, acorde a su distancia de P. La Figura 3.6 muestra el diagrama de la interpolación bicúbica.

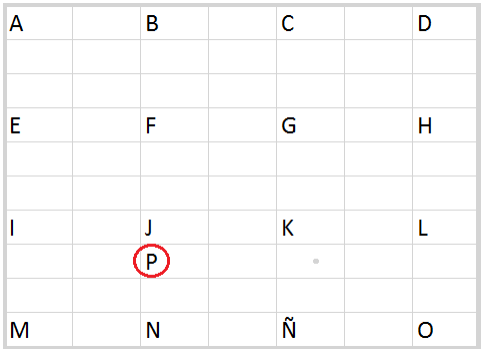


Figura 3.6: Diagrama de la interpolación bicúbica.

En diferencia de la interpolación bilineal, la interpolación bicúbica extiende su influencia por un factor de tres, con lo cual usa métodos avanzados de interpolación para hacer estimaciones entre las coordenadas y el valor de P. El diagrama de flujo, presentado en la Figura 3.7 , es similar al de interpolación bilinear.

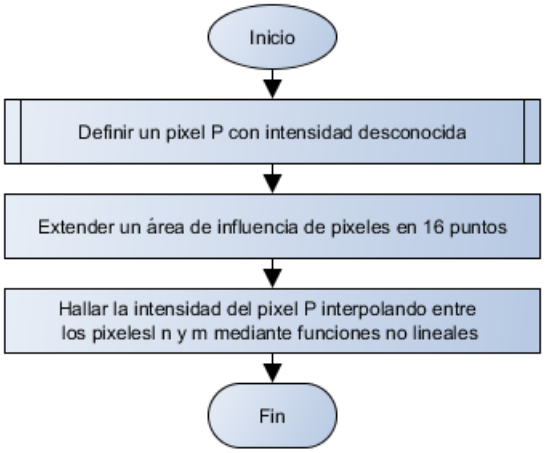


Figura 3.7: Diagrama de flujo de la interpolación bicúbica.

Presentando un ejemplo, para calcular los valores entre A y B, se necesitan usar cuatro valores de pixeles $A, B, A+1, B+1$ para obtener una curva mediante cálculos no lineales.

La complejidad del algoritmo se expresa en la ecuación 3.9

$$\theta(n^3) \tag{3.9}$$

que está en función de las características de la imagen y el cálculo no lineal de los procesos de inteporlación avanzados.

3.4. Métricas de evaluación

Para la evaluación del desempeño de los algoritmos se escogieron tres tipos de mediciones cuantitativas para comparar la calidad en la imagen resultante (Wang y cols., 2004) y el tiempo de procesamiento de los algoritmos.

3.4.1. PSNR

La Señal a Ruido de Pico es la métrica de calidad que usa el cálculo del error medio cuadrado (MSE) y divide el rango máximo del tipo de dato usado por el MSE. Esta medida es simple de calcular pero algunas veces no concuerda con la calidad visual percibida por los humanos. El uso más habitual del PSNR es como medida cuantitativa de la calidad en la reconstrucción de imágenes. Los valores de PSNR se representan en unidades de decibelios (dB); un alto valor de PSNR representa que el error medio cuadrado entre la imagen de referencia y la de trabajo es bajo, lo cual implica que ha sido mejorada adecuadamente. En caso de que el valor sea cero, quiere decir que no hay distorsiones en la imagen y su valor de PSNR tiende a infinito. La representación matemática se muestra en la ecuación (3.10) :

$$PSNR = 20log_{10}(\frac{MAX_f}{\sqrt{MSE}}) \tag{3.10}$$

Donde **MSE** se representa en la ecuación (3.11):

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}\|f(i,j) - g(i,j)\|^2 \tag{3.11}$$

Donde:

F representa la matriz de datos en la imagen original,

g matriz de datos en la imagen degradada en cuestión,

m número de filas de los pixeles de las imágenes e **i** su indexado,

n número de columnas en los pixeles de la imagen y **j** representa su indexado,

$\boldsymbol{MAX_f}$ valor de señal máximo que existe en la imagen original que es considerada la adecuada.

3.4.2. SSIM

La Similitud estructural trabaja mediante la coincidencia de los patrones locales en la intensidad de los pixeles que han sido normalizados por luminiscencia o contraste. El proceso usa la imagen

en cuestión basándose de una referencia que usualmente necesita ser de excelente calidad. Utiliza parámetros que son la luminiscencia, contraste e información estructural entre las dos imágenes a comparar. Esta métrica se basa en el principio de que el sistema visual de los humanos es bueno extrayendo información basado en la estructura de un objeto. Los valores devueltos por SSIM radican en un rango de 0 a 1 continuos, donde entre más cercano esté a uno, más parecida será la imágen saliente con la de referencia. La representación matemática se muestra en la ecuación (3.12).

$$SSIM(X,Y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.12}$$

Donde $\mu_x, \mu_y, \sigma_x, \sigma_y$ y σ_{xy} son las medias locales, desviaciones estándar y covariancias cruzadas para las imágenes x, y.

3.4.3. Tiempo

El tiempo de procesamiento de los algoritmos fue medido con la función **clock** de la libreria en C++ **time.h**, cuyas unidades de medición son retornadas en milisegundos..

Está función devuelve el tiempo consumido por el programa expresado en pitidos de reloj, las cuales son unidades de tiempo en una constante, con un largo específico del sistema en uso. Los resultados llegan a variar en función de las características del procesador.

3.5. Discusión sobre los algoritmos de súper resolución seleccionados

A lo largo de la literatura, el uso de los algoritmos del vecino más cercano, interpolación bilineal e interpolación bicúbica han sido marco de referencia para nuevos métodos de súper resolución debido a la simplicidad que tienen con respecto a su implementación, complejidad computacional y resultados aceptables.

El algoritmo del vecino más cercano es el más sencillo y veloz de los tres seleccionados. Sin embargo, tiende a crear distorsiones significativas en las imágenes junto con artefactos como mosaicos y el fenómeno de diente de sierra. El método por interpolación bilinear posee mayor complejidad que el vecino más cercano, carece de desperfectos en la discontinuidad de la escala de grises de los pixeles y generalmente presenta resultados satisfactorios. Además, dado que este método posee propiedades de filtrado de paso bajo, los componentes de frecuencia alta se desvanecen y el contorno de la imagen tiene cierto grado de opacidad. Por otro lado, el método de interpolación bicúbica tiende a obtener, relativamente, mejor calidad visual en la imagen pero con la desventaja de hacer cálculos más complejos y tardados; lo cual lo hace el algoritmo más lento de los tres. Es común ver el método de interpolación bicúbica implementado en varios programas comerciales de procesamiento de imágenes.

Capítulo 4

Implementación en el sistema embebido Jetson TK1

En este capítulo se introduce la Paralelización de los métodos de súper resolución, que incluye una breve introducción a la historia del cómputo paralelo y el sistema embebido utilizado, información sobre la plataforma CUDA(Computer Unified Device Architecture), la tarjeta de desarrollo Jetson TK1, configuración inicial de la tarjeta, la librería OpenCV4Tegra y por último, la configuración de la GPU tegra para la implementación del sistema de procesamiento de imágenes médicas.

4.1. Introducción

Desde los años 1986 a 2002 la velocidad de la frecuencia de reloj en los microprocesadores se incrementó en un promedio de 50 % (Buell, 2011). Debido a esto, los desarrolladores de software solo tenían que esperar por la siguiente generación de microprocesadores para incrementar el rendimiento de sus aplicaciones. Sin embargo, desde el 2002 el rendimiento de los procesadores de un solo núcleo ha disminuido en un 20 % por año. Debido a esta reducción, el factor en la velocidad de la frecuencia de reloj en los microprocesadores se ha reducido hasta detenerse debido a las limitaciones físicas en los dispositivos. Dado que el rendimiento de los microprocesadores se encontró estancado, en el 2005 la mayoría de los fabricantes de microprocesadores había optado por tomar el paradigma del cómputo paralelo a favor de incrementar el rendimiento de procesamiento. Esto quiere decir que en lugar de seguir diseñando mono-procesadores ahora construirían múltiples procesadores en un solo circuito integrado.

Esto ha provocado un gran cambio para los desarrolladores de software: añadir más procesadores no mejorará de una manera simple el rendimiento de la mayoría de los programas seriales, es decir, programas que fueron escritos para correr en un solo procesador. Tales programas no toman en consideración los múltiples procesadores y por consecuencia, no aprovechan las ventajas de tener múltiples procesadores.

La importancia de atender las nuevas plataformas de hardware que soportan el procesamiento paralelo radica en tres cuestiones:

- Incremento perpetuo de la velocidad de procesamiento.

- Incremento en la densidad de transistores en circuitos integrados.
- Los programas seriales llegan a no ser del todo paralelizables; es necesario comenzar a asimilar modelados de cómputo paralelo.

Por lo tanto, dado que el futuro de la computación de alto desempeño radica en el paralelismo, es de suma importancia el aprendizaje de este nuevo paradigma en beneficio del cómputo de alto desempeño.

4.2. CUDA

En el año de Noviembre de 2006 la compañía Nvidia introdujo lo que es CUDA, una plataforma de cómputo paralelo de propósito general que mediante un modelo de programación, aprovecha las capacidades de cómputo en una GPU Nvidia (NVIDIA, 2017) para resolver problemas complejos de una manera más eficiente que una CPU (Central Processing Unit). CUDA viene en un entorno de desarrollo que permite usar C como lenguaje de programación por defecto. CUDA tiene soporte para otros lenguajes como son:

- C++.
- Fortran.
- Python.
- Java.

El tipo de arquitecturas de GPU que soporta CUDA son:

- Pascal.
- Maxwell.
- Kepler.
- Fermi.

Los origenes de CUDA surgen por la alta demanda de GPU's que soporten el procesamiento en tiempo real de gráficos 3D de alta definición. Ante esto, se diseñaron procesadores que tuvieran las características de ser múltihilo, gran poder de procesamiento y alto índice de banda ancha. CUDA fue creado para habilitar el hardware de las tarjetas en el aprovechamiento en otras aplicaciones fuera del renderizado de gráficos.

Existe una gran diferencia de las capacidades de procesamiento entre una CPU convencional y una GPU. Esto se debe a que los datos del tipo punto flotante son más rápidos de procesar en una GPU debido a que el renderizado de gráficos trabaja, especialmente, con este tipo de datos; por ende las tarjetas gráficas dedican más transistores al procesamiento de datos que a la memoria cache y el control de flujo, en contraste con las CPU's. La Figura 4.1 muestra un diagrama referente a la distribución de memoria en ambos dispositivos.

a b

Figura 4.1: Distribución de memoria en CPU (a) y GPU (b) (NVIDIA, 2017).

De manera concreta, las GPU's están especialmente diseñadas para atender problemas modelados en forma de computación para datos paralelos (un programa siendo ejecutado en muchos elementos de manera paralela, con gran intensidad aritmética). Debido a que el mismo programa es ejecutado en cada elemento de los datos, los requerimientos de control son menores. También, dado que el programa es ejecutado en muchos datos y tiene una alta intensidad aritmética, es posible ocultar la latencia de acceso en memoria con cálculos de procesamiento en lugar de datos usados por la memoria caché.

Los mapas de procesamiento de datos paralelos almacenan los elementos de datos en hilos de procesamiento paralelo. Gracias a esto, muchas aplicaciones que procesen grandes conjuntos de datos son aptos para usar un modelo de programación paralela. En el renderizado 3D se utilizan enormes conjuntos de pixeles y vértices que son mapeados en hilos paralelos. De manera similar, en aplicaciones de procesamiento de imágenes tales como el post-procesado de imágenes renderizadas, codificación y decodificación de vídeo, escalado de imágenes, estéreo visión y reconocimiento de patrones tienen la posibilidad de mapear bloques/ventanas de imágenes y pixeles a procesamientos paralelos. Existen otros algoritmos fuera del campo del renderizado y procesamiento de imágenes que se benefician por el modelo paralelo, que van desde el procesamiento de señales, simulación de fenómenos físicos, finanzas y hasta biología.

4.3. Jetson TK1

La tarjeta de desarrollo Jetson TK1 posee características útiles que son propias de los sistemas embebidos (Cuadra, Alonso, Duran, y Fernando, s.f.). Tales características son:

Confiabilidad: Las fallas no pueden ser resueltas con ayuda del usuario por lo que estos tienden a tener expectativas más altas en lo que se refiere a un sistema robusto. Las consecuencias de una falla pueden ser graves hasta el punto de poner vidas humanas en riesgo (control industrial, aeronáutico, automotriz).

Recursos limitados de hardware: Como son de tamaño reducido en comparación con las computadoras convencionales, los sistemas embebidos tienen memoria limitada, energía limitada y fuentes de energía alternativa relativamente inestables(celdas solares, baterías de litio). Pero, al limitar los recursos se obtiene la gran ventaja de la fácil movilidad del equipo y su portabilidad en agentes móviles como los son robots, drones, entre otras aplicaciones relacionadas.

Respuesta en tiempo real: Las operaciones de software deben ejecutarse en periodos exactos (hard real time) y en periodos limitados (soft real time), por lo tanto, la respuesta que deben tener

el tiempo real demanda altos índices de velocidad en el procesamiento de cómputo, más que en las aplicaciones de computadoras convencionales.

Tales ventajas han resultado valiosas para el desarrollo de una variedad de aplicaciones. Algunas aplicaciones en las que es usada la tarjeta Jetson TK1, vienen incluidas en la siguiente lista pero no están limitadas a ellas: visión por computadora, deep learning, robótica, cómputo en GPU, etc. Muchas de estos desarrollos han construido sistemas de control de navegación de drones, reconocimiento de imágenes médicas, sistemas robóticos autónomos, etc.

La Jetson TK1 de Nvidia es una plataforma de desarrollo embebida de linux potenciada por una tarjeta gráfica **TegraK1**. La tarjeta lleva preinstalado de fabrica un sistema operativo de Nvidia (versión modificada de Ubuntu 14.04), que incluye los controladores necesarios para operar el sistema, pero requiere la instalación de otros paquetes para utilizar las librerías de CUDA, OpenCV, Tensorflow, etc. La Figura 4.2 muestra una imagen de la tarjeta de desarrollo, tal y como viene de fabrica:

Figura 4.2: La tarjeta de desarrollo Jetson TK1 de Nvidia.

Las características técnicas se mencionan en la Tabla 4.1, (más información en (*Jetson TK1 - eLinux.org*, 2017)):

4.4. OpenCV4tegra

El paquete de visión por computadora de código abierto OpenCV ((*NVIDIA GameWorks Documentation - Introduction*, 2014)) es una colección de librerías nativas en C/C++ que contienen más de 2500 funciones de procesamiento de imágenes y visión por computadora. El paquete ha estado disponible al público desde hace más de una década y ha sido usado en aplicaciones de código abierto y comerciales. Con una comunidad de más de 50.000 desarrolladores alrededor del mundo, OpenCV soporta varios sistemas operativos de escritorio y móviles incluyendo Windows, Linux, Max OS X, Android e iOS, siendo ejecutado en arquitecturas de x86, MIPS y ARM. Entre sus características se encuentra soportada la plataforma CUDA en GPU's de escritorio y dispositivos móviles. La naturaleza múlti-plataforma de OpenCV es ventajosa para compilar aplicaciones de visión por computadora en diferentes plataformas con, relativamente poca dificultad.

Tabla 4.1: Características principales de la Jetson TK1

Dimensiones	127m x 127mm
Procesador Tegra K1 SOC	**GPU**: Nvidia Kepler "GK20a"
	CPUNvidia "4-plus"2.32GHZ ARM quad-core
DRAM	2GB DDR3L
Almacenamiento interno	16GB
Puertos perifericos	USB 3.0 Tipo A hembra y 2.0 micro AB hembra,
	HDMI
	Puerto serial DB9 RS232
	Audio ALC5639 Realtek HD Audio codec
	Ethernet RTL8111GS Realtek
	SATA 2.5 y 3.5
	JTAG para depuración profesional
	GPIO x 7 pins a 1.8V
API's soportados	CUDA 6.0
	OpenGL 4.4
	OpenMAX IL multimedia codec
	NPP (NVIDIA Performance Primitives)
	OpenCVTegra
	VisionWorks
Alimentación	12V DC

OpenCV for Tegra (OpenCV4Tegra) es una versión de OpenCV para android que ha sido optimizada por Nvidia, específicamente para la familia Tegra. Está disponible en los sistemas operativos Android y Vibrante (Linux embebido). Contiene optimizaciones específicas de Tegra, permitiendo a OpenCV ejecutar aplicaciones aún más rápidas que la implementaciones de OpenCV para Android.

4.5. Computación paralela en GPU

En está sección se introducen los conceptos principales del modelo de programación en CUDA los cuales son Kernels, jerarquía de hilos, jerarquía de memoria, programación heterogénea y capacidad de cómputo (NVIDIA, 2017).

CUDA permite a los programadores definir funciones, definidas como Kernels, que al ser llamadas se ejecutan un número de n veces en paralelo por n hilos diferentes en CUDA, distribuyendo el trabajo en múltiples procesadores, siendo ésta la diferencia en la programación secuencial.

Un Kernel se define usando la decla ración **__global__** junto con el número de hilos que ejecutará ese kernel dando una llamada dada por una sentencia de configuración de ejecución <<<...>>>. Cada hilo que ejecuta el kernel se le asigna un índice único nombrado por *thread ID*, que es accesible por el kernel mediante la variable constructor **threadIdx**.

La jerarquía de los hilos, **threadIdx** se representa en un vector de tres componentes que identifican a los hilos en forma de bloques de uno, dos o tres dimensiones llamados *thread block*. Esto provee una manera natural de llamar funciones con con dimensiones tales como el vector, matriz y

volumén (x, y, z).

La relación de los hilos (thread ID) con sus índices es la siguiente: para un bloque uni-dimensional, el índice es el mismo; para un bloque bi-dimensional (Dx, Dy) el hilo del índice (x,y) es (x+y Dx); para un bloque tridi-mensional de tamaño (Dx, Dy, Dz) el hilo del índice (x,y,z) es (x + y Dx + z Dx Dy). Existe un limite al número de hilos por bloques, dado que todos los hilos de un bloque residen en el núcleo del procesador y deben compartir los recursos limitados de memoria. En GPU's actuales un bloque de hilos llega a contener hasta 1024 hilos. Aún así, un kernel posee la característica de ser utilizado por múltiples bloques de hilos de un mismo tamaño, es decir, el número total de hilos es igual al número de hilos por bloque muliplicado por el número de bloques. Los bloques están organizados en marcos de una, dos o tres dimensiones, como se puede observar en la Figura 4.3.

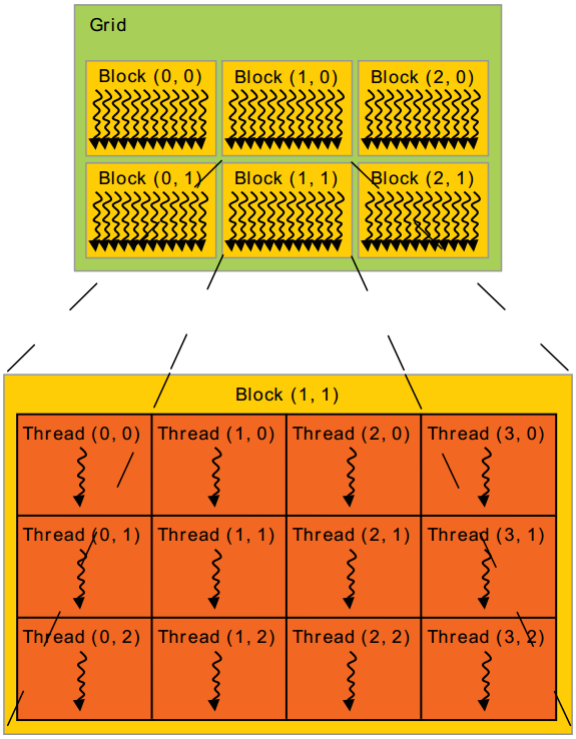


Figura 4.3: Organización de los bloques en CUDA, donde muestra como el marco (Grid) está compuesto de bloques (Block) y este de hilos (Thread), (NVIDIA,2017) .

El número de bloques de hilos se determina, generalmente, por el tamaño de los datos que van a ser procesados o el número de procesadores en el sistema. El número de hilos por bloque y el número de bloques por marco se especifican en la sintaxis <<< ... >>> que llega a ser de tipo **int** o **dim3**. Cada bloque dentro del marco se identifica por un índice de una, dos o tres dimensiones, que es accesible en el kernel por el constructor **blockIdx**. La dimensión del bloque de hilos es accesible a traves del constructor **blockDim**.

Los hilos dentro de un bloque trabajan en conjunto con datos de la memoria compartida y sincronizando su ejecución a accesos de memoria coordinados; se sincronizan de manera específica los puntos de sincronización en el kernel mediante la función intrínseca **__syncthreads()**, que actua como una pausa por la que todos los hilos en el bloque deben esperar antes de ejecutar algún

proceso.

En la jerarquía de memoria, los hilos tienden a tener acceso a los datos de múltiples espacios de memoria durante su ejecución; cada hilo tiene una localidad de memoria privada. Cada bloque de hilos tiene una memoria compartida que es visible para todos los hilos que comparten un mismo tiempo de ejecución en el bloque.

Existen dos espacios en memoria adicionales de solo lectura, que son accesibles para todos los hilos: el espacio de memoria constante y de texturas. Los espacios globales, constante y de texturas están optimizados para diferentes tipos de usos. Por ejemplo, el espacio en memoria de textura ofrece diferentes modos de direccionamiento y filtrado de datos para específicos formatos. En los kernel usados por la misma aplicación, los espacios asignados globales, constantes y de texturas, persisten para su uso.

La programación heterogénea hace referencia al concepto de usar programación secuencial convencional y paralela en una misma aplicación. El modelo de programación CUDA hace uso de las siguientes abstracciones: las operaciones del **host** (anfitrión) y las de **device** (dispositivo). El host y device hacen referencia a los procesadores de la máquina; la unidad de procesamiento central (CPU) y la unidad de procesamiento gráfico (GPU), los cuales trabajan de manera colaborativa en una relación de esclavo-maestro, siendo maestro el CPU. Esto se refiere a que todo el procesamiento paralelo (ejecutado por los hilos de los procesadores en la GPU) va a ser comandado por el procesador anfitrión. Device trabaja para host.

Todos los hilos se ejecutan desde device, que opera como un coprocesador del host. Los kernel se ejecutan en la GPU (device) y el resto del programa en el CPU (host). El host y device mantienen espacios de memorias separados en la DRAM (Dynamic ramdom acces memory-Memoria dinámica de acceso aleatorio), siendo apuntadas como memoria del host y memoria del device. Por ende, un programa que maneje los espacios de memoria globales, constantes y texturas son visibles a los kernel mediante llamadas de rutinas en CUDA. Esto incluye la asignación de memoria y transferencia de datos entre la memoria del host y device.

4.6. Configuración del sistema embebido

Para desarrollar proyectos en la Jetson TK1 es necesario instalar el paquete de software **JetPackL4T** (*NVIDIA Embedded Systems*, 2014). La JetpackL4T es un conjunto de paqueterías que incluye todo lo necesario para trabajar con librerías como CUDA, Vision Works, OpenCV, etc. También incluye lo que es:

- Deep Neural Network library (cuDNN).
- TensorRT.
- Multimedia API.
- Tegra Graphics Debugger.

Antes de comenzar con el proceso de configuración de la tarjeta, es prioritario revisar que la tarjeta Jetson TK1 funcione correctamente. Para eso se realiza una comprobación de los controladores (drivers) de Nvidia que vienen ya incrustados de fabrica en la versión L4T. Los requisitos básicos para hacer la comprobación y realizar trabajos en la plataforma embebida son:

- Una tarjeta Jetson TK1.
- Tener conectado los respectivos puertos periféricos:
  - USB HUB con 4 puertos conectados a los dispositivos periféricos de entrada (mouse/teclado).
  - Cable HDMI conectado al dispositivo externo de despliegue (display).

La tarjeta debe conectarse a un monitor con soporte HDMI (High definition media interface-Interfaz multimedia de alta definición) para la visualización del sistema. También debe conectarse a un teclado y un mouse mediante un USB hub.

Una vez que la tarjeta ha sido establecida en el área de trabajo, se prosigue a conectar el adaptador de corriente AC a una toma de 120-240V, desde donde se conectará al puerto de alimentación DC de 12v en la tarjeta. El sistema deberá encenderse, de lo contrario presione el botón *power*. Un LED (Light emitting diode-diodo emisor de luz) verde se encenderá junto con el ventilador disipador de calor, lo cual indica que el sistema ha sido iniciado correctamente.

En la pantalla deberá observarse la línea de comando en Linux. Para acceder al sistema como administrador las credenciales son:

- **Usuario:** ubuntu
- **Contraseña:** ubuntu

Hecho esto, es imperativo activar la **interfaz gráfica** para hacer fácil el siguiente procedimiento, que es habilitar la interfaz gráfica del sistema operativo. Para eso hay que activar los controladores especiales de Nvidia que vienen en la tarjeta mediante las siguientes instrucciones:

1-En la línea de comando, teclear:

CD$ {HOME}/NVIDIA-INSTALLER

para dirigirnos al directorio donde se encuentran los controladores (HOME es el respectivo directorio).

2- Teclear en la línea de comandos:

sudo ./installer.sh

Este comando extraerá e instalará los controladores de Linux. El sistema se reiniciará y la próxima vez que se inicie sesión, será desde la interfaz gráfica de Gnome. El proceso de instalación de los controladores solo hay que hacerlo una vez.

Terminado esto, se prosigue a instalar las paqueterías de Jetpack para dar habilitar el entorno de desarrollo para aplicaciones. La instalación requiere una computadora de escritorio que tenga como sistema operativo **Ubuntu Linux x64 (14.04)** a la que se hará referencia como huésped. También es necesario una conexión a internet y como mínimo, 10 GB de espacio en almacenamiento.

En la Jetson TK1 solo se requiere que un cable USB micro-B este conectado a la computadora huésped. Es necesario que la tarjeta Jetson esté conectada a internet (una forma sencilla es mediante conexión ethernet).

Con los requisitos anteriores cubiertos, el proceso de instalación está listo para iniciar.

En la computadora huésped se debe tener el archivo Jetpack (que es descargado en línea desde la página oficial de Nvidia (*NVIDIA Embedded Systems*, 2014)). Al momento de escribir esta guía, se está usando la versión 2.3.1, en 64 bits de Ubuntu.

Ya descargado el archivo Jetpack, se debe proceder a crear una carpeta con el nombre **Jetson** en el directorio de **carpeta personal**. El propósito de nombrar las carpetas es ofrecer una fácil ubicación y organización de los archivos que van a ser descargados. Posteriormente, los archivos Jetpack son movidos a la carpeta Jetson.

Se abre la terminal de comandos y se teclea:

cd Jetson/

Se traslada automáticamente a la carpeta Jetson para teclear:

/s

Una vez hecho esto, para comprobar que los archivos se encuentren ahí, teclear:

chmod tx Jetpack-L4T-2.3.1-linux-64.run

deberá aparecer una ventana para la instalación. NOTA: Es posible que aparezca una ventana de advertencia, como se muestra en la Figura 4.4.

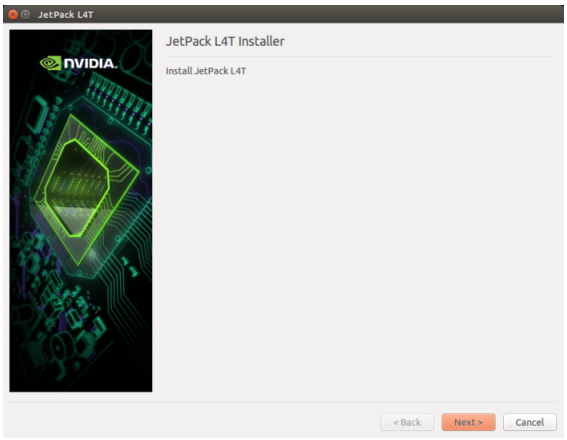


Figura 4.4: Proceso de instalación-Inicio.

Dar click en Next>y se mostrará la siguiente dirección donde se instalarán los componentes:

/home/user/Jetson/

Dar click en Next>para solicitar que se especifique las tarjetas a trabajar y descargar sus respectivos componentes, como se muestra en la Figura 4.5

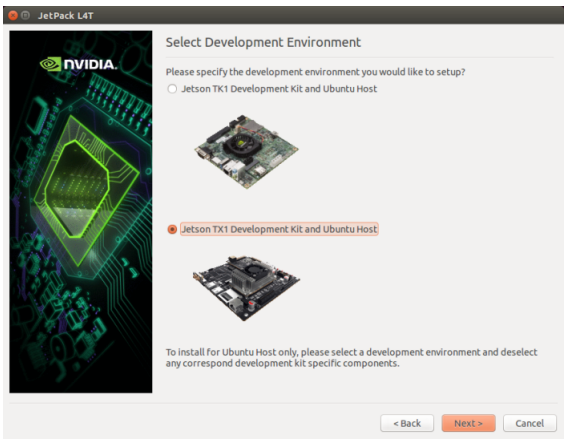


Figura 4.5: Proceso de instalación-Selección del hardware.

Se selecciona la Jetson TK1 y se da click en Next>. El programa hará ciertas descargas y luego mostrará otra ventana en pantalla donde se gestionan los componentes que serán descargados, tal y como se ve en la Figura 4.6.

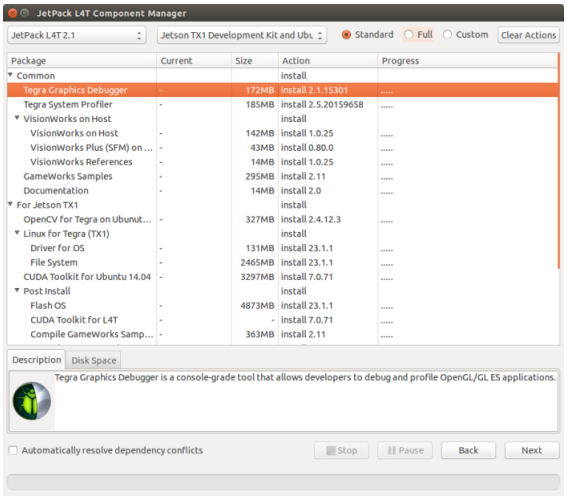


Figura 4.6: Proceso de instalación-Gestión de componentes.

Los paquetes a destacar son: Linux For Tegra (TK1), Flash OS, CUDA Toolkit for L4T y OpenCVforTegra. Una vez seleccionados los componentes se da click en Next>, se aceptan los términos de uso para cada paquete y una vez más dar click en Next>. Con eso empezará el proceso de descarga e instalación.

Nota: en el proceso quizá se requiera de validación de permisos por parte del usuario, por lo que es aconsejable, supervisar el proceso entero.

La descarga e instalación tardará varios minutos, dependiendo de la conexión a internet. Una vez terminado de instalar/descargar todo en la computadora huésped, aparecerá una ventana mencionando que se ha completado el proceso. Dar click en Next>.

En esta fase, la computadora huésped procederá a instalar los componentes seleccionados pre-

viamente del Jetpack a la Jetson TK1. Nota: el medio de conexión a la computadora depende de la configuración previa de los componentes seleccionados.

Para esta fase de la instalación se dividen dos procedimientos en función de si se habilitó la característica de **Flash OS**. Sí no ha sido seleccionado en el gestionador de componentes, se necesitará añadir una dirección IP, nombre de usuario y contraseña para establecer una conexión ssh al dispositivo objetivo. En la Figura 4.7 se aprecia la ventana emergente.

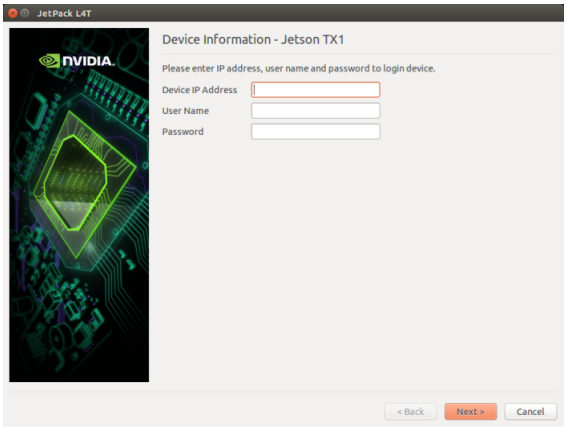


Figura 4.7: Proceso de instalación-Gestionador de Flash OS.

Se introduce la información pertinente y se hace click en Next>, por lo que Jetpack iniciará con la instalación de sus componentes en el dispositivo. Por el caso contrario, se necesitará una topología de red, como se muestra en la Figura 4.8:

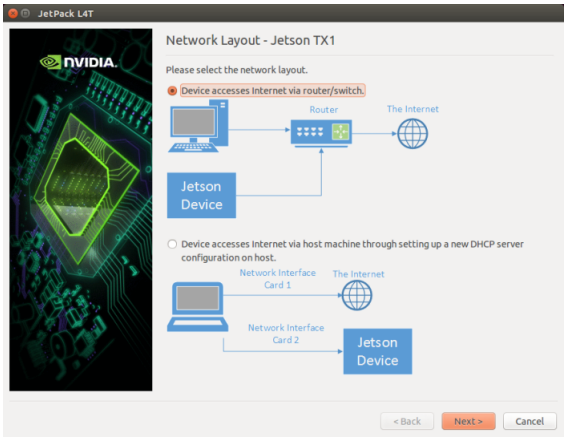


Figura 4.8: Proceso de instalación-Topología de red.

Ya terminado este proceso aparecerá otra ventana pidiendo el modo de conexión en la red. Hay dos opciones de comunicación: Mediante **Device acces Internet via router/switch** y **Device get IP assigned by DHCP server on host and access Internet via host machine**, para esta guía se eligió la comunicación mediante router. La ventana de selección deberá ser similar a la Figura 4.9:

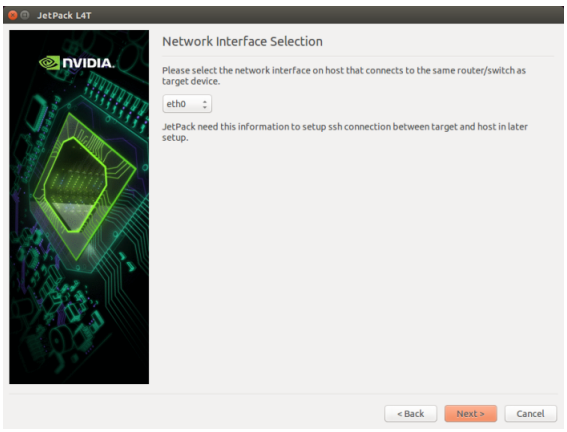


Figura 4.9: Proceso de instalación-Interfaz de la red.

Dado que es más fácil comunicar la computadora huésped a la Jetson TK1, se selecciona la opción .^{et}h0". Dar click en Next>y aparecerá una ventana con el resumen de todo. Volver a dar click en Next>y aparecerá una ventana de línea de comando, pidiendo permisos para configurar la Jetson en modo de recuperación. En este momento se debe montar la tarjeta y prepararla.

Para poner la Jetson TK1 en modo de recuperación se debe:

1. Asegurarse de que el dispositivo está apagado.
2. Conectar el cable micro-B (incluido en el paquete de fabrica) a su respectivo puerto en la Jetson, el otro extremo debe conectarse al puerto USB de la computadora huésped.
3. Conectar el adaptador de corriente a la tarjeta.
4. Presionar el botón power para encender la tarjeta.
5. Presionar y mantener el botón **FORCE RECOVERY**, mientras se presiona y libera el botón **RESET**.
6. Esperar dos segundos y liberar el botón **FORCE RECOVERY**.
7. Mientras la tarjeta esté en modo de recuperación, teclear en la terminal del huésped:

   lsusb

La tarjeta deberá aparecer como:

Bus 006 Device 006 : ID 0955: 7140 Nvidia corp

Este mensaje significa que la comunicación entre el huésped y la tarjeta fue exitosa. En la línea de comandos del huésped, se teclea *return* para continuar con la instalación. Una vez terminado, se da click en el botón *finish* y con eso se habrá completado el proceso de instalación de los paquetes Jetpack L4T en la Jetson.

Paso importante antes de conectar la Jetson a internet.

Antes de conectar la tarjeta a internet se debe prevenir que el gestionador de paquetes .ªpt-emplace el archivo "libglx.so.ªl momento de realizar las actualizaciones del sistema. Para evitarlo, se da el siguiente comando en terminal:

sudo apte-mark hold xserver-xorg-core

La librería es específica de los controladores de Nvidia. Si esta fuera ser remplazada por un controlador de Ubuntu, provocará fallas al momento de cargar la interfaz gráfica. De ahí su importancia de mantenerlo intacto.

Finalmente, es recomendable cambiar el nombre y contraseña de usuario de fabrica, ya que estos son estándar por lo que, al tener acceso a internet, la seguridad del sistema puede verse comprometida.

4.7. Configuración de la GPU

La configuración utilizada para la GPU Tegra K1 constó del uso de ventanas de procesamiento, usando hilos para procesar cada pixel de la imagen. Los hilos fueron agrupados en bloques para crear las ventas con el propósito de recorrer las dimensiones de la imagen entera y aplicar los algoritmos de súper resolución.

Se tomaron las imágenes de resonancia magnética resultantes con una resolución de 812x812. Los hilos han sido agrupados de manera bidimensional en 16x16 por cada bloque, siendo en total 256 hilos por bloque. Cada pixel es tratado por un hilo dedicado. En la Tabla 4.2 se enlistan los parámetros de la GPU Tegra K1.

Tabla 4.2: Parámetros de la GPU Tegra K1.

Núcleos de CUDA	192
Número de múltiprocesadores	1
Número máximo de hilos por bloque	1024
Memoria compartida	4.9 Kb
Librería del módulo CUDA en OpenCV	*warping*

El entorno de desarrollo constó de:

- Sistema operativo Linux4Tegra, versión 3.10.40.
- Índice de reloj del GPU: 0.85 Ghz.
- Ancho del bus de memoria: 924 Mhz.

Considerando la continuidad de los datos y los bordes de la imagen, se cargan 16x16 pixeles en los hilos a la memoria compartida, usando 512 Bytes. Sin embargo, el tipo de mapeo con respecto a los pixeles no está aprovechando en su totalidad el paralelizado de los bloques debido al proceso serial que se realiza en los bordes de la imagen; esto se debe a que la resolución de la imagen es 812 x 812 y no existe en su dominio un número múltiplo de 32 (en CUDA los hilos se agrupan en una unidad mínima de 32 elementos llamada warp) para distribuir apropiadamente los bloques, por lo tanto, no se aprovecha en su totalidad el procesamiento paralelo de la GPU. Como observación,r

un análisis más detallado con respecto al modelado paralelo de los algoritmos es ideal para futuros trabajos que planteen optimizar el proceso de paralelización.

Capítulo 5

Experimentación y análisis de resultados

En este capítulo se presenta la experimentación y análisis de resultados del sistema, se divide en las secciones de introducción, metodología de experimentación, entorno de pruebas y el análisis sobre los resultados obtenidos; se discuten las mejoras observables y se realiza pruebas de comprobación en los resultados.

5.1. Bases de datos

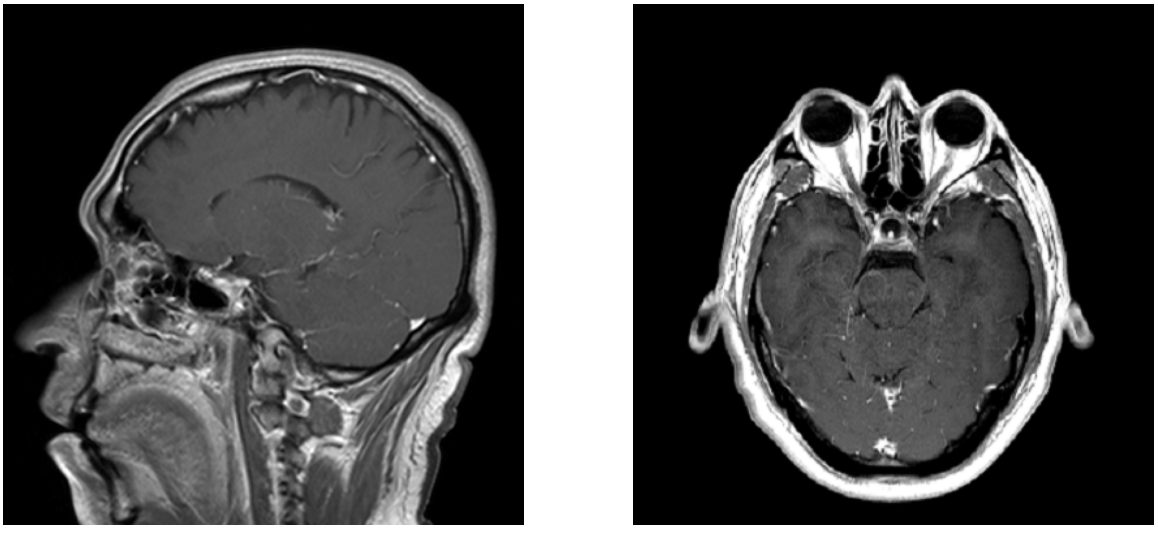
Esta sección presenta información breve sobre las imágenes de resonancia magnética usadas para las pruebas, las bases de datos de las que fueron obtenidas y cuyas fuentes son tanto privadas como públicas.

5.1.1. Bases de datos obtenidas

Se recolectaron cuatro estudios de imágenes de resonancia magnética: tres privadas y una pública. Los estudios privados fueron proporcionados por el Instituto Tlaxcalteca de Asistencia Especializada a la Salud (ITAES, ver Figura 5.1), por cortesía del médico especialista en radiología Adriana Carmona. Los estudios contenían 609 imágenes del cráneo; 272 de la rodilla y 316 de la columna vertebral. Todas las imágenes son presentadas en el formato DICOM, con vistas sagital, coronal y axial. Las imágenes tienen una resolución de 212 x 212 a 356 x 356, en función del tipo de vista. En total, son 1197 imágenes disponibles de una base de datos privada.

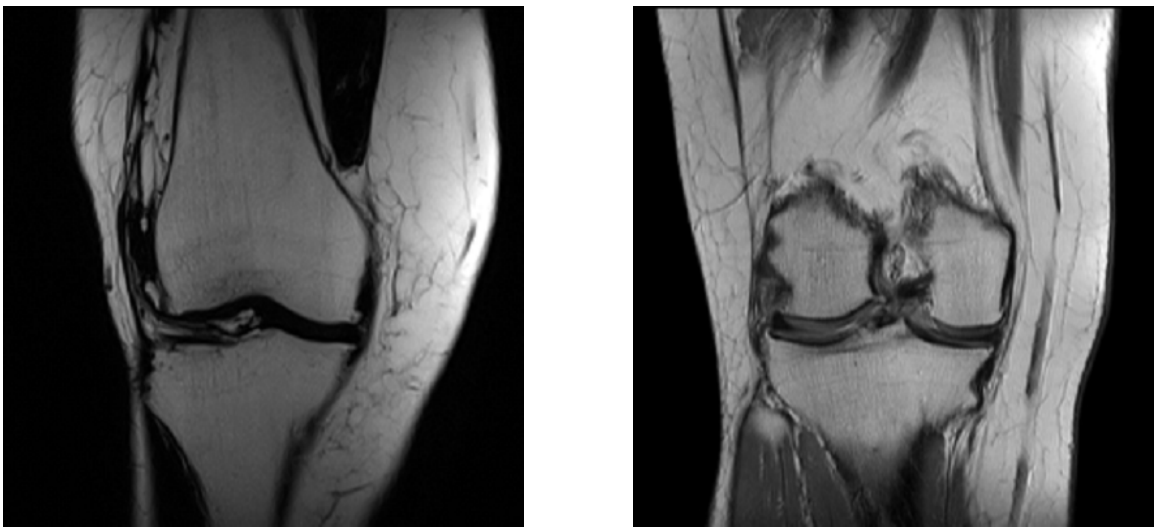
Figura 5.1: Instituto Tlaxcalteca de Asistencia Especializada en la Salud (ITAES), donde fueron recolectadas las imágenes de fuentes privadas.

Dichas imágenes fueron otorgadas bajo estrictas condiciones sobre la confidencialidad de los datos personales de los pacientes, siendo guardadas en tres CD-ROM en posesión del departamento de ciencias computacionales del Cenidet. En las Figuras 5.2, 5.3 y 5.4 se aprecian muestran de los estudios.



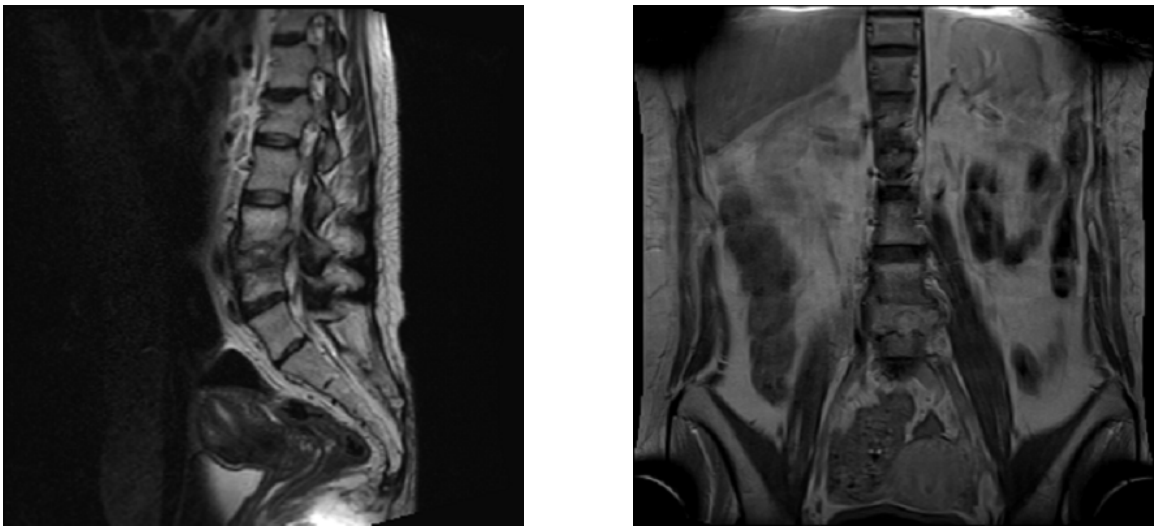
(a) Vista sagital. (b) Vista axial.

Figura 5.2: Imágenes del cráneo.



(a) Vista sagital. (b) Vista coronal.

Figura 5.3: Imágenes de la rodilla.



(a) Vista sagital. (b) Vista coronal.

Figura 5.4: Imágenes de la columna vertebral.

El estudio público fue descargado de una base de datos llamada Çancer imaging archive"(referenciada en el apéndice A.4), identificada con el nombre de RIDER Neuro MRI"(Rider), teniendo un número de 1508 imágenes en formato DICOM, con rangos de resolución de 128 x 128 a 256 x 256, mostrando únicamente el plano sagital. Una muestra de las imágenes se aprecia en la Figura 5.5 :

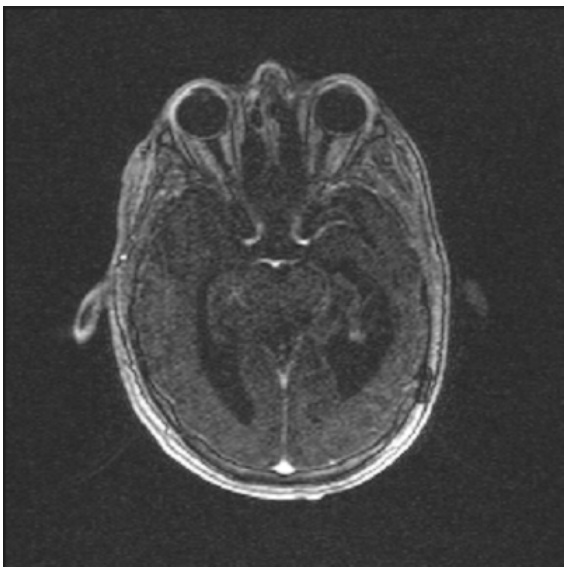


Figura 5.5: Imagen de la base de datos Rider.

5.1.2. Justificación

El propósito de tener dos bases de datos con fuentes públicas y privadas reside en garantizar la obtención de resultados concretos en las evaluaciones al no usar solo casos de prueba aislados, de tal forma que se obtenga un conocimiento más amplio sobre el desempeño de los algoritmos clásicos de SR en diferentes casos de estudio.

5.2. Metodología de experimentación

5.2.1. Entorno de desarrollo

Para la sesión de experimentos, donde se evaluaron los algoritmos clásicos de súper resolución, se cuenta con el siguiente entorno de desarrollo:

- PC Laptop con un procesador AMD A8-6410 APU de 2GHz, 8 Gb de memoria RAM, Sistema operativo Windows 8 x64.
- IDE Eclipse Mars 2.0.
- Compilador MinGW 5.1.0.
- Opencv 3.0.1.
- MicroDicom 2.0.0.

5.2.2. Preprocesamiento

MicroDicom (*Microdicom viewer*, s.f.) es utilizado para cambiar el formato DICOM a .bmp en los cuatro estudios, realizando una homogeneización de la resolución a 400x400, siendo en total 2706 imágenes pre-procesadas.

Este proceso ayuda a normalizar los elementos del muestreo y obtener datos más limpios. Además, para la fase de evaluación es necesaria, por los parámetros en las métricas cuantitativas, una imagen de referencia con la misma resolución, por lo que homogeneizar las imágenes es crucial para realizar mediciones.

Sin embargo, es notable mencionar que los parámetros de las métricas cuantitativas tienen valores fijos basados en los estándares utilizados por la comunidad de desarrolladores de OpenCV (OpenCV_Community, 2017).

5.3. Entorno de pruebas

La metodología general para el plan de experimentación se muestra en la Figura 5.6, donde los procesos son descritos de la siguiente manera:

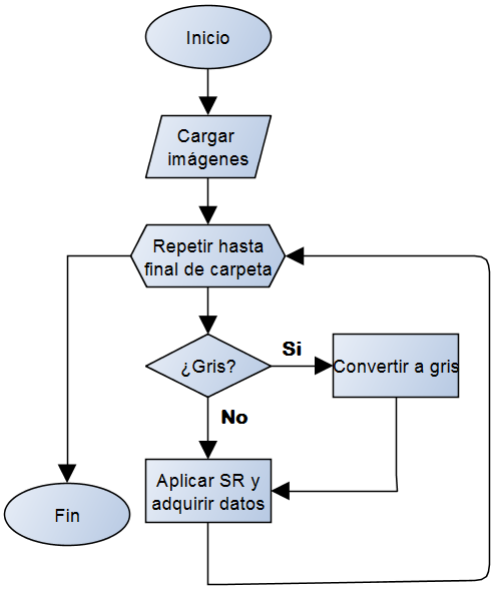


Figura 5.6: Diagrama de metodología.

- **Cargar imágenes:** se cargan las imágenes en una carpeta entera para realizar un procesamiento por lotes.
- **Repetir hasta final de carpeta:** el ciclo principal inicia, donde se trabajarán todas las imágenes de una carpeta.
- **¿Gris?**:Módulo opcional, en caso de que las imágenes vengan a color. Realiza la conversión de color a escala de grises mediante una fórmula generalmente aceptada para tratar la lumi-

niscencia de los pixeles en un sistema RGB. La fórmula se expresa en la ecuación 5.1:

$$Y = \frac{0{,}2989R + 0{,}5870G + 0{,}1148B}{3} \tag{5.1}$$

Donde Y es el valor de intensidad en grises que tomará el pixel y R, y B son los valores de intensidad en rojo, verde y azul.

Este módulo quedo sobrante de la fase de pruebas del sistema y, aunque no es necesario para las MRI, se optó por dejarlo para futuras pruebas en el sistema que requiera el uso de imágenes a color .

- **Aplicar SR y adquirir datos:** se utilizan los tres algoritmos de interpolación junto con la adquisición de datos (más adelante se muestran detalles al respecto).

El proceso del módulo Aplicar SR y adquirir datos se muestra más a detalle en la Figura 5.7. Los procesos se describen como:.

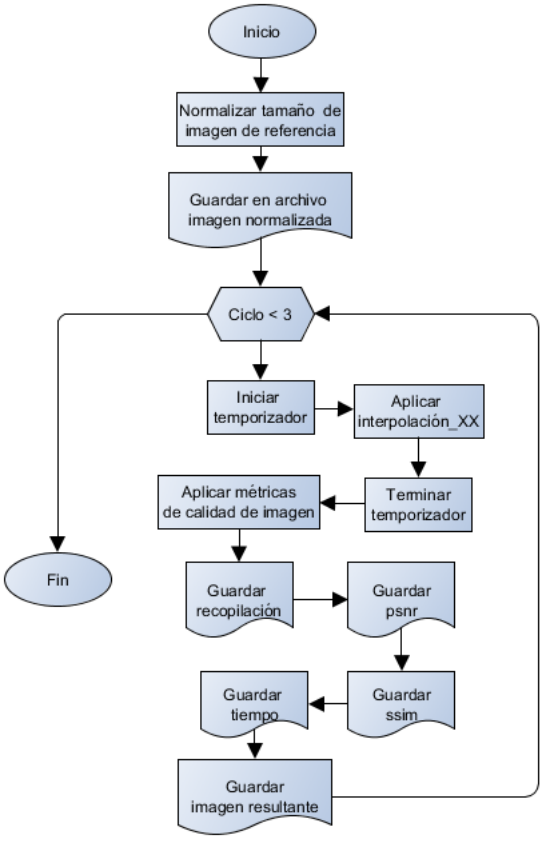


Figura 5.7: Metodología de la aplicación de los algoritmos de SR.

- **Normalizar la resolución de imagen de referencia:** esta imagen será utilizada como referencia para la comparación. Se imprime en disco duro.

La imagen de referencia fue creada a partir del algoritmo de remuestreo en el área de los pixeles (Parker, Kenyon, y Troxel, 1983),(Derenyi y Saleh, 1989). Que fue seleccionado debido a su uso previo en la literatura, que precede a los algoritmos clásicos seleccionados.

- **Ciclo <3:** ciclo donde se probarán los algoritmos de vecino más cercano, interpolación bilineal y bicúbica. El ciclo inicia con un valor de 0 para llevar un control del uso en los algoritmos de interpolación.

  0 == vecino más cercano | 1 == Interpolación bilineal | 2 == Interpolación bicúbica

- **Iniciar y terminar temporizador:**Medición del tiempo mediante la función clock.
- **Aplicar interpolación_XX:** Aplicación de los algoritmos de súper resolución. Se usa uno a la vez por ciclo.
- **Aplicar métricas de calidad de imagen:** Las métricas PSNR y SSIM son utilizadas midiendo la imagen de referencia con la aumentada.
- **Escritura de datos:** procesos donde almacenan en un archivo de texto la información obtenida.
- **Guardar imagen resultante:** Se guarda en disco la imagen mejorada por los algoritmos de súper resolución.

Todas las imágenes son aumentadas a una resolución de 812x812 pixeles utilizando los tres algoritmos clásicos de súper resolución. Los datos obtenidos de las métricas cuantitativas y las imágenes muestra, son almacenados en el disco duro para su posterior análisis.

5.4. Resultados de las evaluaciones

En esta sección se presentan los resultados obtenidos de las evaluaciones, tanto para la implementación en CPU como en GPU, utilizando las métricas de calidad seleccionadas y el tiempo de procesamiento.

5.4.1. Resultados en CPU

Los resultados obtenidos son presentados en valores de PSNR y SSIM. Para identificar la calidad que representan las métricas se debe comprender los conceptos en la lectura de sus valores de medición:

PSNR devuelve una señal de error, medida en unidades de decibelios (dB), proveniente de la imagen de referencia en función de la imagen evaluada; entre más pequeño sea el valor de retorno, menor es la discrepancia entre las dos imágenes. Por lo tanto, valores mayores equivalen a una imagen más semejante con respecto a la referencia.

Por otro lado, SSIM tiene un rango de valores continuos entre 0 y 1, dominio que determina la similitud estructural que tiene una imagen a la referencia. Valores que estén más cercanos a 1 muestran mayor similitud. Los valores de 0 en PSNR y 1 en SSIM quieren decir que la imagen evaluada es exactamente igual a la de referencia.

Calidad de imagen

Los resultados obtenidos de las pruebas se muestran en las Figuras 5.8 a 5.15, con los algoritmos siendo expresados como NN para vecino más cercano, BL para interpolación bilineal y BC para interpolación bicúbica. Las unidades en el eje horizontal (y) están en función en decibelios para PSNR, en cuanto a SSIM las unidades están en un valor continúo de 0 a 1.

La Figura 5.8 muestra los resultados promediados de 609 imágenes en PSNR de los estudios del cráneo. Se observa que la interpolación bilineal es la que posee el valor más alto de error, lo que significa que ha preservado mejor calidad después del aumento de las imágenes.

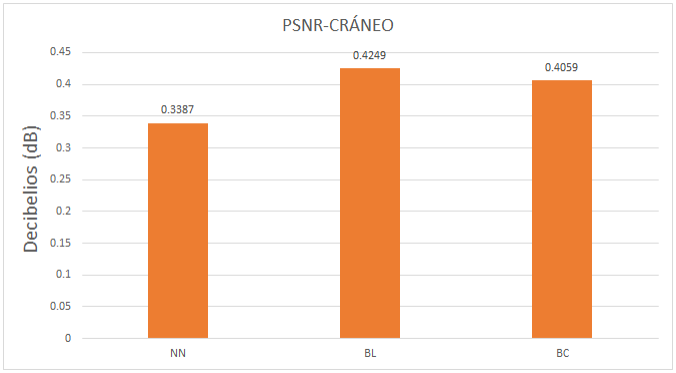


Figura 5.8: PSNR de los estudios del cráneo.

De manera similar, la Figura 5.9 presenta resultados promediados de 272 imágenes de los estudios en la rodilla, que se asemejan a los estudios del cráneo, no en valor cuantitativo si no en orden de jerarquía en calidad de los algoritmos.

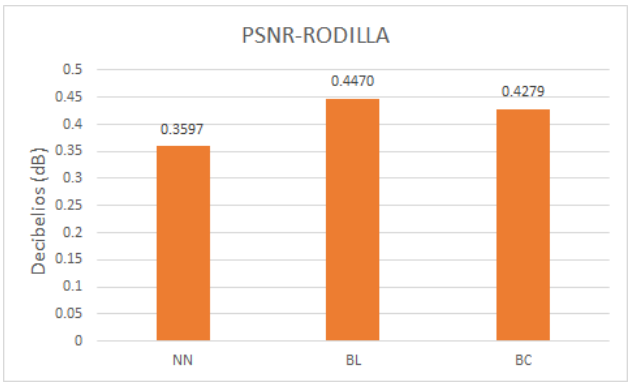


Figura 5.9: PSNR de los estudios de la rodilla.

En la Figura 5.10 se llega a la misma observación en los resultados promediados de 316 imágenes de los estudios de la columna vertebral. La interpolación bilineal supera a los otros dos algoritmos en cuestión de calidad de imágen en los tres estudios de fuentes privadas utilizadas, según la métrica PSNR.

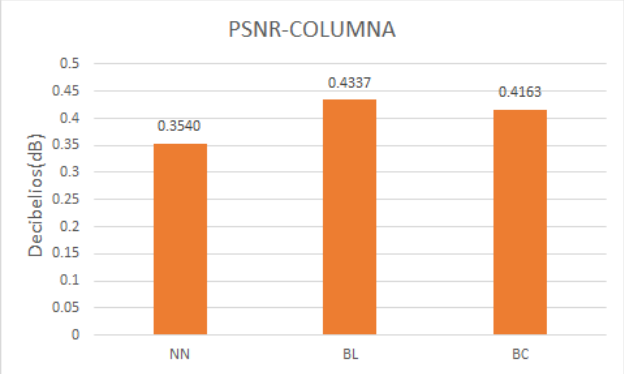


Figura 5.10: PSNR de los estudios de la columna.

De la evaluación realizada en los resultados promediados de las 1197 imágenes de la base de datos pública Ryder, mostrada en la Figura 5.11, se llego a la misma conclusión que los resultados anteriores; el algoritmo de interpolación bilineal sigue en la cabeza en cuestión de calidad de imagen.

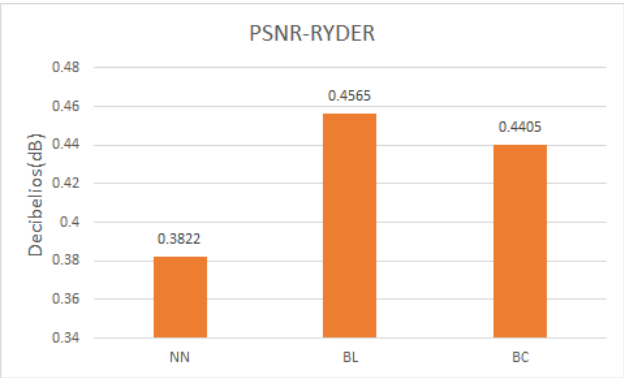


Figura 5.11: PSNR de los estudios Ryder.

La pruebas realizadas en la métrica de SSIM en los estudios del cráneo, mostrados en la Figura 5.12, muestran que el algoritmo de interpolación bilineal es superior en cuestión de calidad de imagen. Una interesante observación es que los valores del algoritmo de interpolación bicúbica tienen un margen de diferencia menor a la interpolación binileal, a comparación de lo obtenido en PSNR.

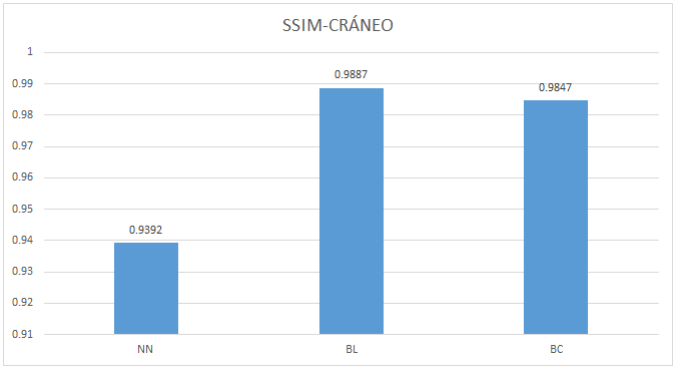


Figura 5.12: SSIM de los estudios del cráneo.

Los valores de la rodilla en la Figura 5.13 se mantienen similares a los estudios del cráneo. Aún con un menor número de muestreo, la diferencia entre el desempeño de los algoritmos se mantiene.

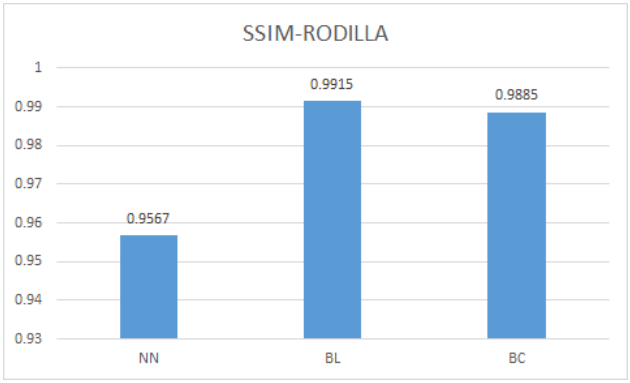


Figura 5.13: SSIM de los estudios de la rodilla.

Finalmente, los valores de la Figura 5.14 muestran que los estudios de la columna mantienen las observaciones de las gráficas anteriores, sin mostrar cambios en la jerarquía por términos de calidad de imagen.

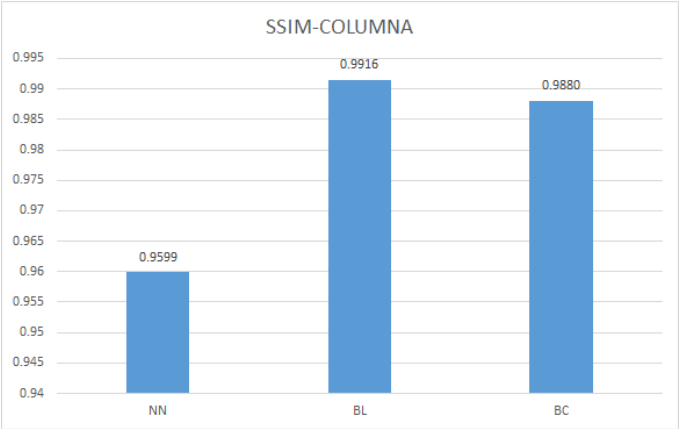


Figura 5.14: SSIM de los estudios de la columna.

Los valores obtenidos en los estudios Ryder, mostrados en la Figura 5.15, mantienen las conclusiones obtenidas en los anteriores estudios. Es notable que, la diferencia entre la interpolación bilineal y la bicúbica es menor ahora que ha sido evaluado con una mayor cantidad de imágenes. Sin embargo, la interpolación bilineal se mantienen a la delantera.

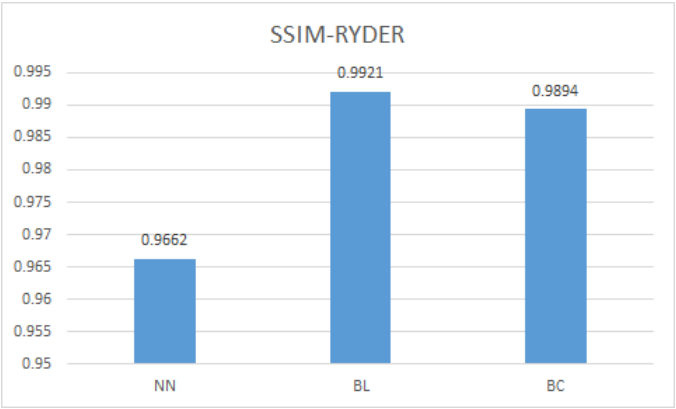


Figura 5.15: SSIM de los estudios Ryder.

Tiempo de procesamiento

El tiempo de procesamiento ha sido medido en unidades de milisegundos, ignorando las rutinas previas y solo tomando en cuenta el tiempo donde se procesan los algoritmos. Las Figuras 5.16 a 5.19 muestran el desempeño en tiempo de los algoritmos.

La Figura 5.16 muestra el tiempo de procesamiento realizado en los estudios del cráneo, midiendo los tres algoritmos de súper resolución. Se observa que el algoritmo de interpolación bicúbica es el más lento y el vecino más cercano el más rápido.

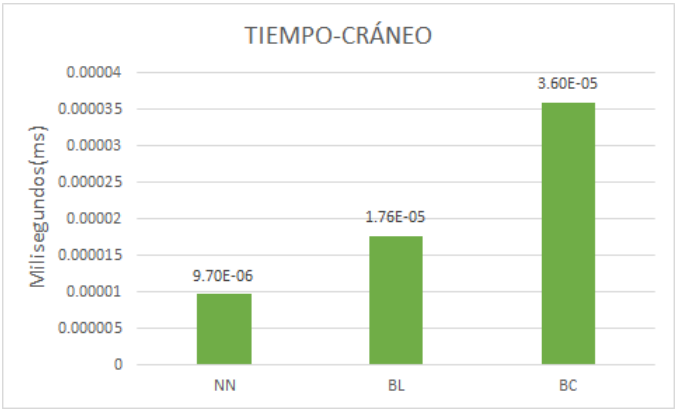


Figura 5.16: Tiempo de procesamiento en los estudios del cráneo.

La Figura 5.17 muestra los resultados en los estudios de la rodilla. Las observaciones se mantienen sin cambios con respecto a lo evaluado en los estudios anteriores, pero se hace notar que el tiempo de procesamiento fue menor en los tres algoritmos. Sin embargo, las muestras de estos estudios son menores que en el anterior por lo que la representación es menos objetiva.

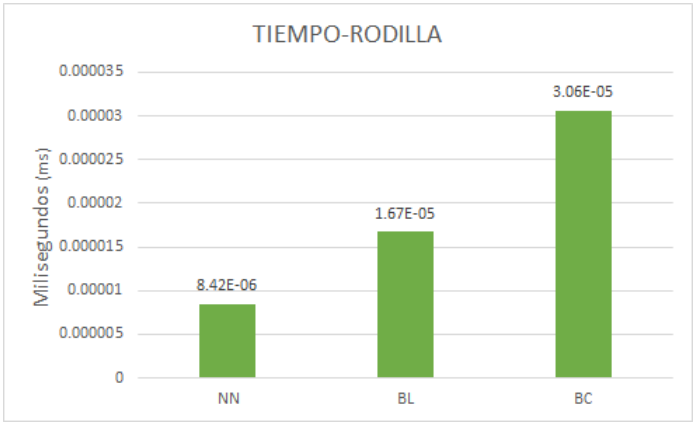


Figura 5.17: Tiempo de procesamiento en los estudios de la rodilla.

Por último, la Figura 5.18 muestra los resultados obtenidos de la columna. Aunque las observaciones de jerarquía se mantienen, este estudio fue el que se tardo más tiempo de procesamiento, considerando que el estudio del cráneo tiene un número mayor de imágenes. La naturaleza entre los objetos representados de las imágenes es una de las causas de esta discrepancia.

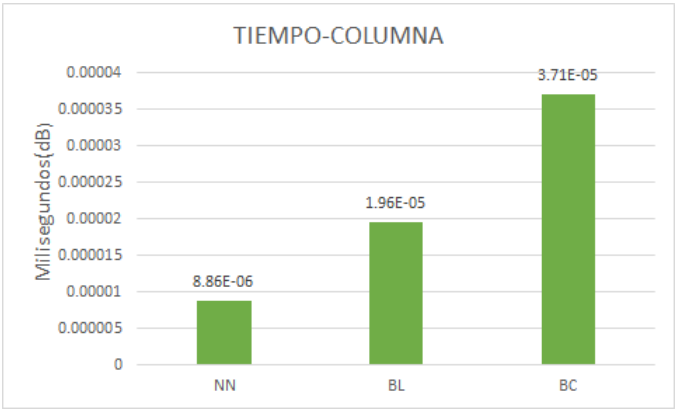


Figura 5.18: Tiempo de procesamiento en los estudios de la columna.

El estudio público Ryder, mostrado en la Figura 5.19, expresa que aún a mayor número de muestras y viniendo de una base de datos pública, los resultados en cuestión de tiempo de procesamiento se mantienen.

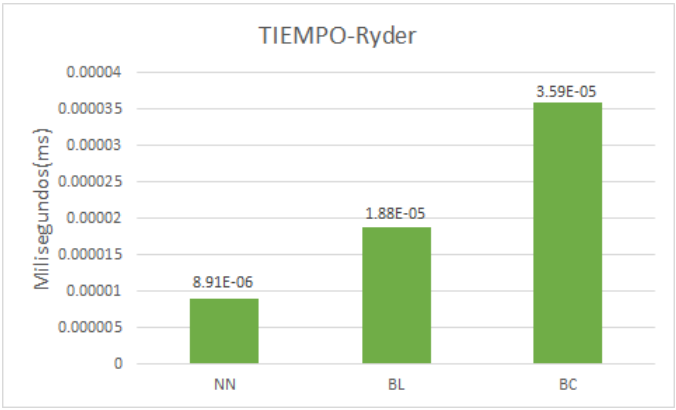


Figura 5.19: Tiempo de procesamiento en los estudios de Ryder.

Resumen de los resultados

Un resumen de los resultados se aprecia en las Tablas 5.1 a 5.4 de los cuatro estudios:

Tabla 5.1: Estudios del cráneo.

Métrica	NN	BL	BC
PSNR	33.8	42.48	40.58
SSIM	0.9392	0.9886	0.9847
Tiempo	97ms	176ms	359ms

Tabla 5.2: Resultados de la Rodilla.

Métrica	NN	BL	BC
PSNR	35.97	44.69	42.79
SSIM	0.9567	0.9915	0.9884
Tiempo	84ms	167ms	306ms

Tabla 5.3: Resultados de la columna.

Métrica	NN	BL	BC
PSNR	35.39	43.36	41.62
SSIM	0.9599	0.9915	0.9880
Tiempo	88ms	196ms	370ms

Tabla 5.4: Resultados de los estudios Ryder.

Métrica	NN	BL	BC
PSNR	38.21	45.65	44.04
SSIM	0.9662	0.9921	0.9893
Tiempo	89ms	187ms	358ms

Discusión

Las tablas muestran los valores de los resultados variaciones con respecto al tipo de estudios que se utiliza. De la base de dato privada, el estudio de la rodilla obtiene un mejor resultado (promedio) con respecto a los otros dos casos.

Se considera que las variaciones en los resultados obtenidos de ambas métricas, están relacionadas con el número de imágenes utilizadas en las evaluaciones: el estudio de la rodilla contiene 272 (teniendo valores más altos en los criterios de evaluación), el de la columna vertebral contiene 316 y el cráneo 609. Las imágenes de la rodilla tienen los promedios más altos mientras que las del cráneo son los menores. La conclusión de esto sería que, dado a la menor cantidad de muestreo en los estudios de la rodilla, sus resultados no aportan una objetividad mayor a los de la columna vertebral y el cráneo.

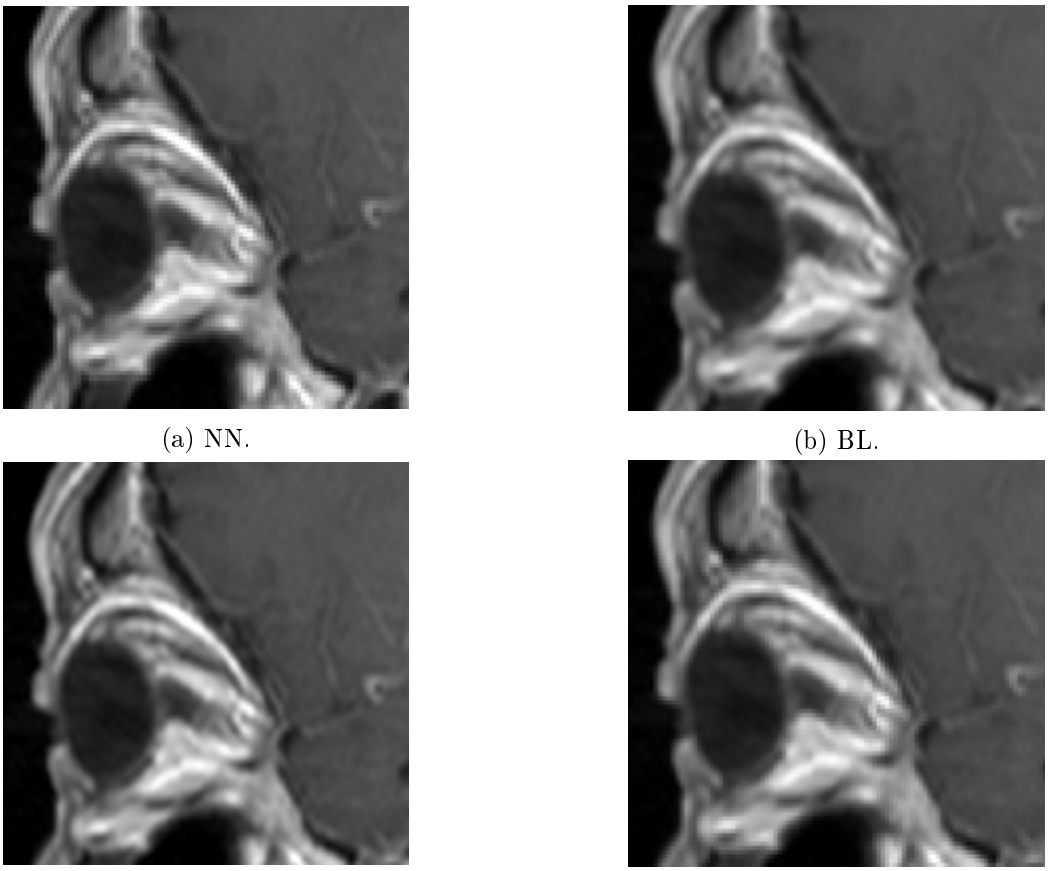
Con respecto a la base de datos pública Ryder, en promedio se obtuvo mejores resultados que con la base de datos privada con respecto a las tres métricas.

Como se puede ver, en la Tabla 5.4 el tiempo de procesamiento fue de 89ms, sin embargo es importante recordar que este repositorio se integra de 1197 imágenes.Con respecto a las métricas de calidad, se puede observar que las variaciones en los resultados no yace, hasta cierto punto, en el número de muestras si no en las propiedades de las imágenes en sí.

Imágenes mejoradas

En las Figuras 5.20 a 5.23 se observa una segmentación de las imágenes resultantes, después de haber sido procesadas con los tres algoritmos de súper resolución, para apreciar los detalles en sus texturas después del aumento:

En la Figura 5.20 se aprecian ciertas diferencias en los estudios del cráneo. En (a) se observa el mejor detalle con respecto al suavizado de las texturas, en comparación con las otras imágenes. En (b) y (c) casi no se llegan a apreciar las diferencias (estas solo llegan a notarse en los resultados cuantitativos).

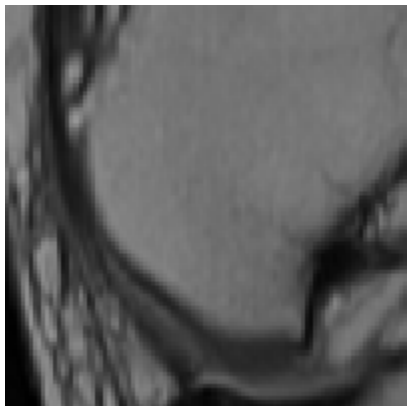


(a) NN. (b) BL.

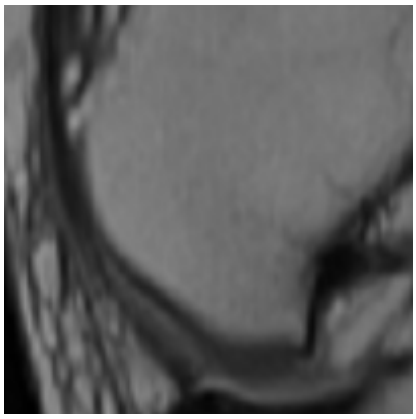
(c) BC. (d) Referencia.

Figura 5.20: Comparación de las imágenes en los estudios del cráneo.

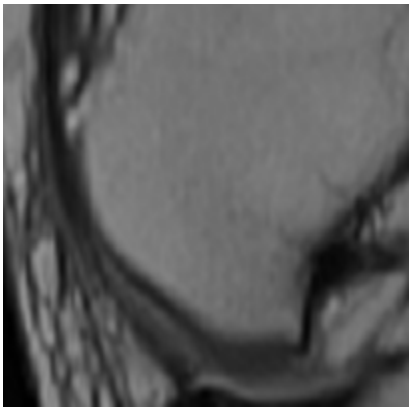
En las segmentaciones de la rodilla, mostrados en la Figura 5.21, se puede apreciar mejor la diferencia de texturas que tiene con respecto a los estudios del cráneo. La materia blanda (representada en pixeles grices) presenta mejor los cambios en sus bordes y texturas.



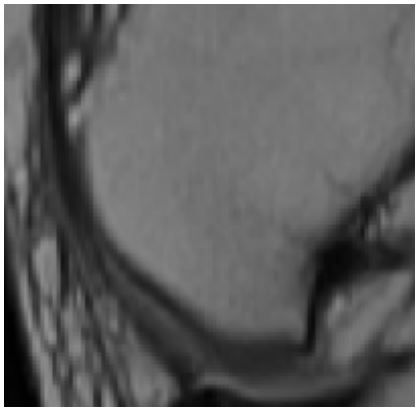
(a) NN.



(b) BL.



(c) BC.

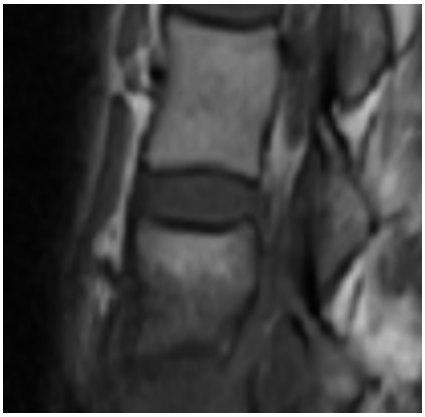


(d) Referencia.

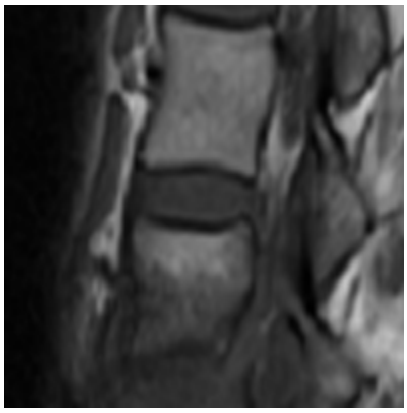
Figura 5.21: Comparación de las imágenes en los estudios de la rodilla.

Finalmente, en las muestras de la columna la diferencia en los bordes se aprecia mejor en las regiones de las vertebras, donde el suavizado de los pixeles define de una forma mas nítida ésta región, aunque sea más difícil de notar debido a la poca claridad de los pixeles representando los huesos. Las imágenes se observan en la Figura 5.22.

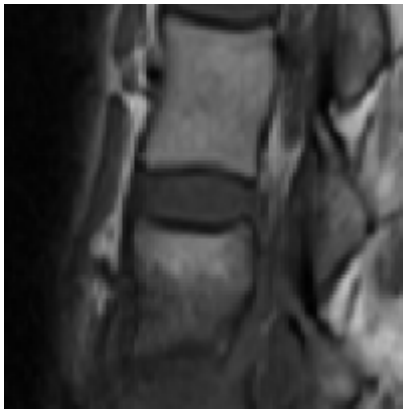
(a) NN.



(b) BL.



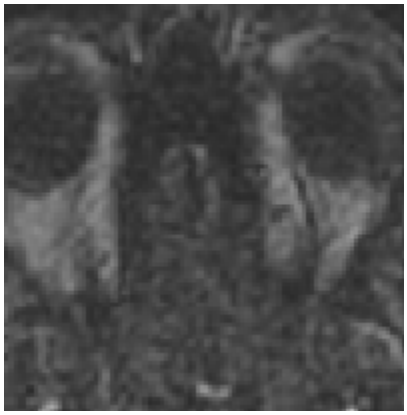
(c) BC.



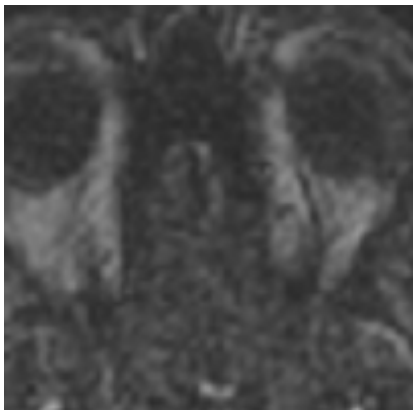
(d) Referencia.

Figura 5.22: Comparación de las imágenes en los estudios de la columna.

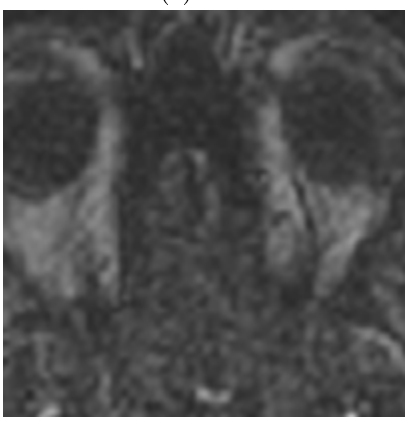
En los estudios ryder (ver Figura 5.23) la imagen es más difícil de visualizar que las anteriores (debido a la calidad original de la imagen) pero, aún así se visualiza que las partes sólidas como el hueso presentan bordes más definidos en (b) y (c) que en (a). Las diferencias en (b) y (c) no llegan a ser visibles al ojo humano pero gracias a las métricas cuantitativas, se estima que el algoritmo de interpolación bilineal se desempeña mejor que el de interpolación bicúbica.



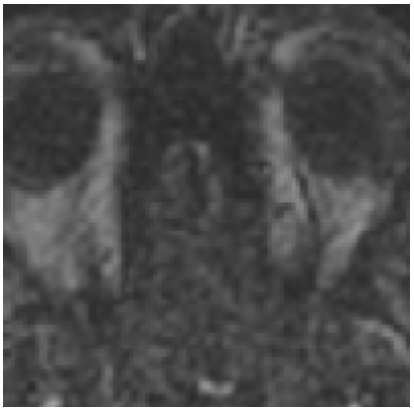
(a) NN.



(b) BL.



(c) BC.



(d) Referencia.

Figura 5.23: Comparación de las imágenes en los estudios de Ryder.

5.4.2. Resultados en GPU

En la plataforma embebida con GPU se implementó el sistema de procesamiento con el mismo plan de pruebas y unidades para el eje horizontal (y). Los resultados se muestran en las Figuras 5.24 a 5.31:

Calidad de imagen

Las gráficas mostradas en la Figura 5.24 reflejan resultados promediados diferentes, utilizando 609 imágenes del estudio del cráneo, a los de la implementación en CPU. Mientras que los valores del algoritmo de interpolación bicúbica y vecino más cercano mantienen su jerarquía, la interpolación

bilineal logró un valor de 0 que quiere decir que la imagen aumentada es exactamente igual a la de referencia.

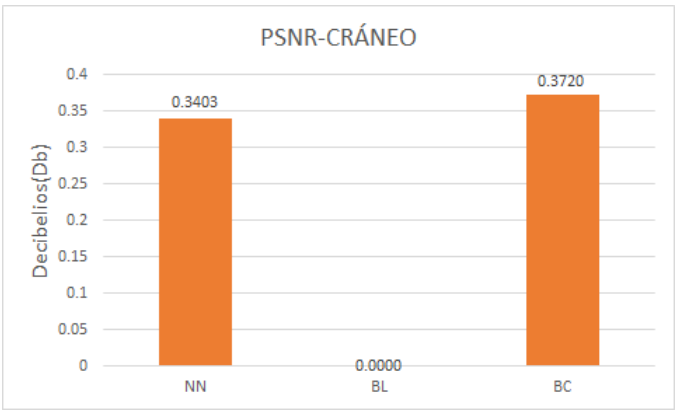


Figura 5.24: PSNR de los estudios del cráneo.

Los estudios de la rodilla, utilizando 272 imágenes, de la Figura 5.25 muestran las mismas observaciones que los estudios del cráneo.

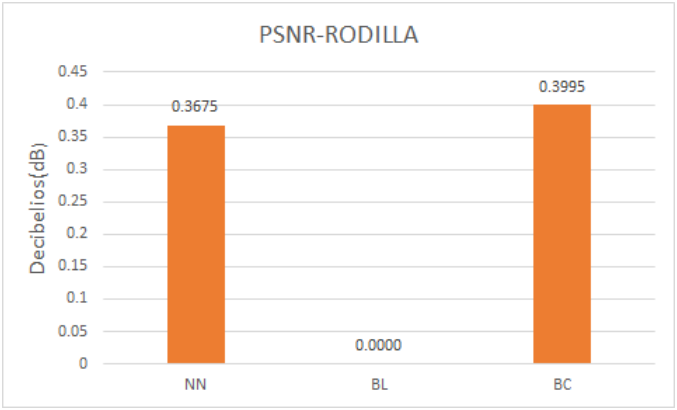


Figura 5.25: PSNR de los estudios de la rodilla.

Por último, los estudios de la columna, utilizando 316 imágenes, de la Figura 5.26, concuerdan con los valores obtenidos de los estudios anteriores.

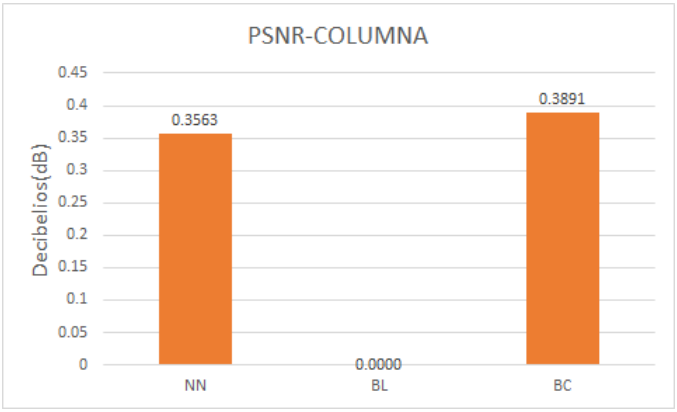


Figura 5.26: PSNR de los estudios del columna.

Para los estudios Ryder, con 1197 imágenes utilizadas, de la Figura 5.27, se observan los mismos resultados en cuestión de la jerarquía de los valores en los algoritmos de súper resolución; la interpolación bilineal obtiene una optimización de sus valores en cuestión de calidad.

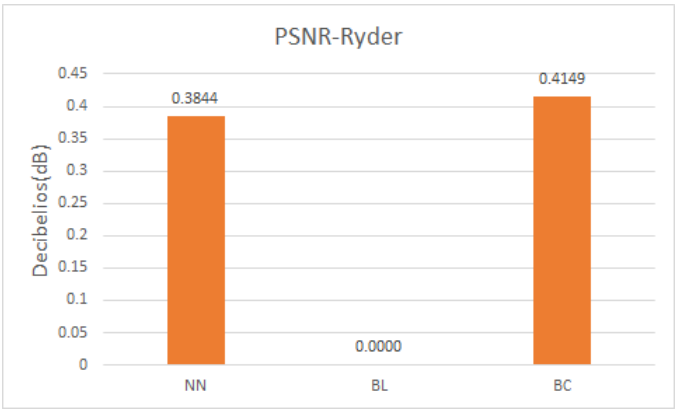


Figura 5.27: PSNR de los estudios de Ryder.

Los resultados obtenidos de los estudios del cráneo en la Figura 5.28 presentan las mismas observaciones que en las métrica de PSNR; el algoritmo de interpolación bilineal llega al valor 1, es decir que la imagen aumentada es igual a la de referencia. El algoritmo del vecino más cercano se desempeño peor en SSIM que en PSNR. La interpolación bicúbica se mantiene como el segundo mejor algoritmo en términos de reconstrucción.

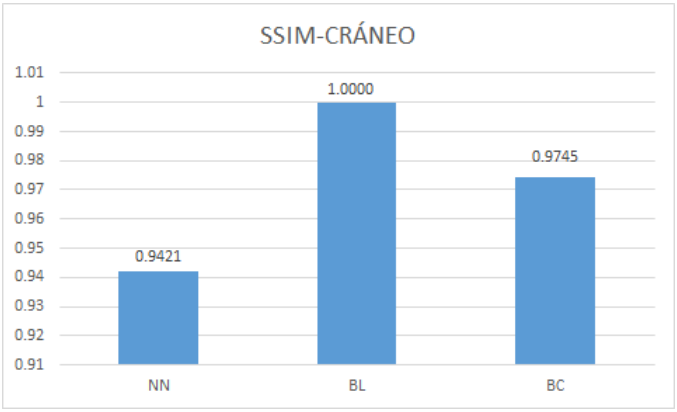


Figura 5.28: SSIM de los estudios del cráneo.

Los estudios de la rodilla en la Figura 5.29 mantienen las mismas observaciones realizadas que en los estudios del cráneo.

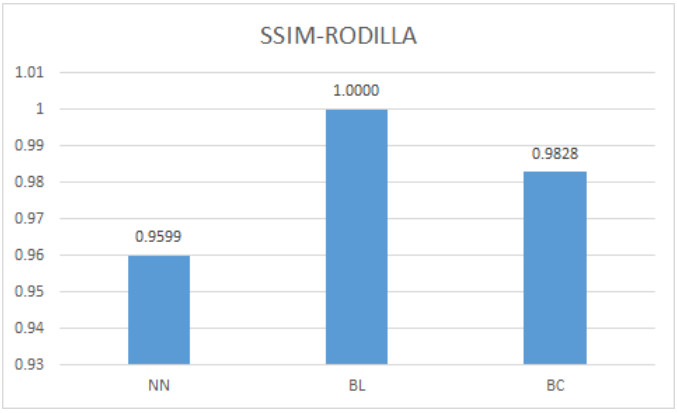


Figura 5.29: SSIM de los estudios de la rodilla.

Finalmente, en la Figura 5.30 se presentan los resultados de la columna, manteniendo la semejanza con los estudios anteriores.

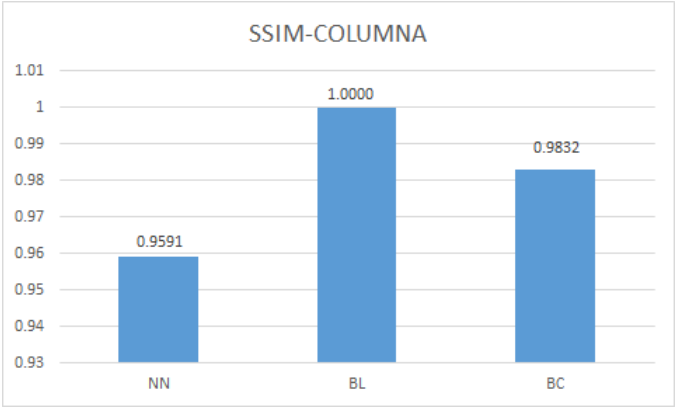


Figura 5.30: SSIM de los estudios de la columna.

Para los estudios Ryder, provenientes de una fuente pública, se presentan los valores obtenidos de SSIM en la Figura 5.31. La jerarquía de los valores en cuestión de calidad se mantiene aún después de examinar un mayor número de muestras que en los otros tres estudios privados. La interpolación bilineal mantiene su rango de valor optimizado y la interpolación bicúbica sigue teniendo mejor valor que la del vecino más cercano.

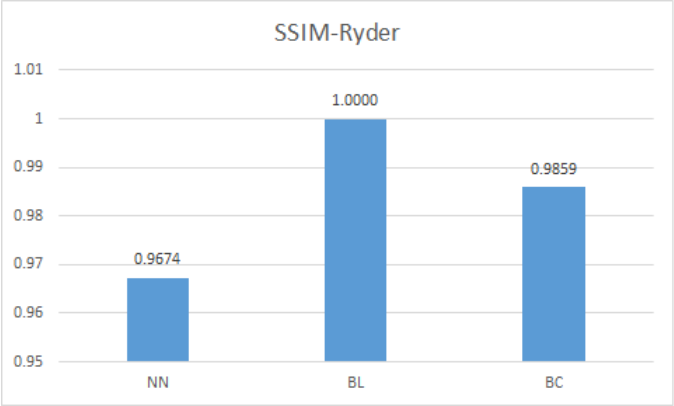


Figura 5.31: SSIM de los estudios de Ryder.

Tiempo de procesamiento

En las gráficas de las Figuras 5.32 a 5.35 se muestra los resultados del tiempo de procesamiento en GPU, donde las unidades se mantienen en milisegundos.

En la Figura 5.32 se aprecian los valores del tiempo de procesamiento en los estudios del cráneo. Los resultados muestran que los valores muestran un margen de diferencia aún menor que en las implementaciones en CPU, sin embargo la jerarquía aún así se mantiene siendo el vecino más cercano el más rápido, la interpolación bicúbica el más lento y la bilineal estando en el medio.

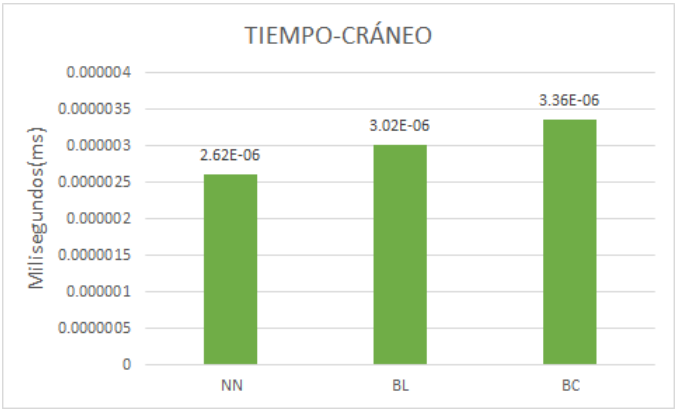


Figura 5.32: Tiempo de procesamiento de los estudios del cráneo.

Los resultados de la rodilla (vistos en la Figura 5.33) preservan los observado en los estudios del cráneo.

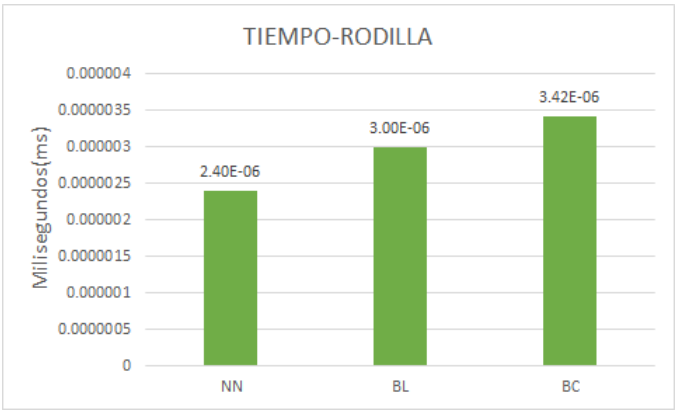


Figura 5.33: Tiempo de procesamiento de los estudios de la rodilla.

Por último, en los estudios de la columna en la Figura 5.34 no se observan cambios en los tiempos de procesamiento, aunque el estudio de la columna es la que menos imágenes tiene, siendo su muestreo menos objetivo que los anteriores.

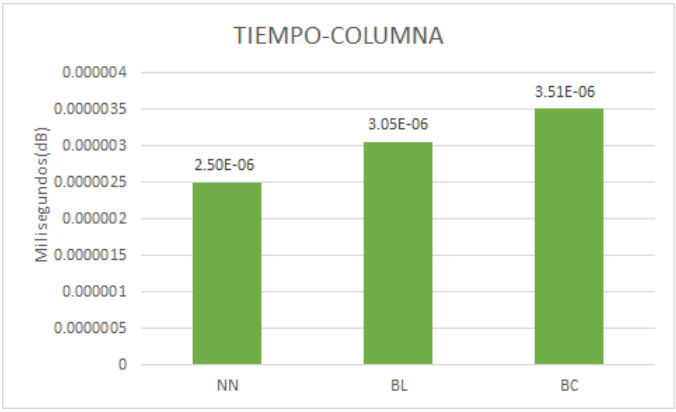


Figura 5.34: Tiempo de procesamiento de los estudios de la columna.

Para la base de datos pública Ryder, en la Figura 5.35 se presentan los valores del tiempo de procesamiento. que mantienen la jerarquía de datos observada en los estudios anteriores. Cabe mencionar que el estudio Ryder es el que tiene mayor número de imágenes en las bases de datos recolectadas, por lo tanto sus resultados aportan mayor objetividad en cuestión de muestreo.

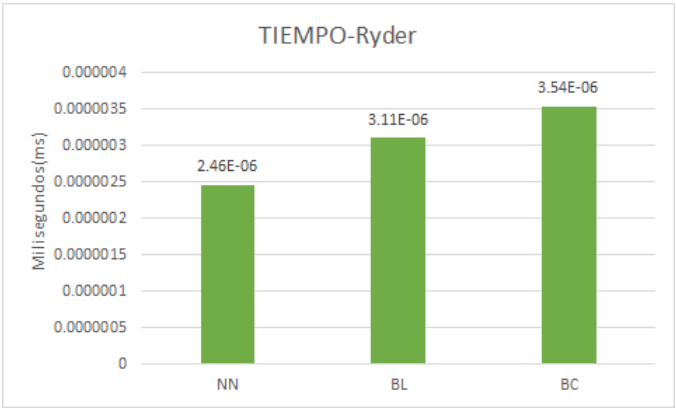


Figura 5.35: Tiempo de procesamiento de los estudios Ryder.

Resumen de los resultados

El resumen de los valores obtenidos son mostrados en las Tablas 5.5 a 5.8:

Tabla 5.5: Estudios del Cráneo.

Métrica	NN	BL	BC
PSNR	34.3	0	37.20
SSIM	0.94	1	0.97
Tiempo	26ms	30ms	36ms

Tabla 5.6: Estudios de la Columna.

Métrica	NN	BL	BC
PSNR	35.62	0	38.90
SSIM	0.95	1	0.98
Tiempo	24ms	30ms	35ms

Tabla 5.7: Estudios de la Rodilla.

Métrica	NN	BL	BC
PSNR	36.74	0	39.94
SSIM	0.95	1	0.98
Tiempo	23ms	30ms	34ms

Tabla 5.8: Estudios Ryder.

Métrica	NN	BL	BC
PSNR	38.44	0	41.48
SSIM	0.96	1	0.98
Tiempo	24ms	31ms	35ms

Discusión

Como se puede observar en las tablas anteriores, los resultados obtenidos en GPU se mantienen con respecto a los logrados en la evaluación en CPU.

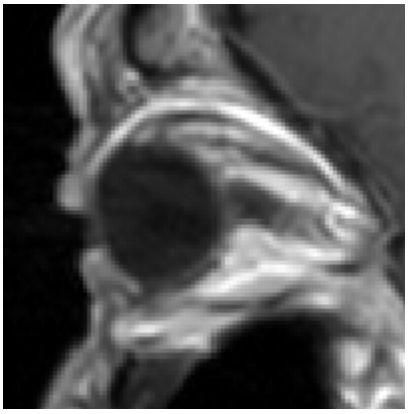
Llama la atención que en los valores obtenidos con el algoritmo de interpolación bilineal se logra una optimización en la implementación del GPU; las métricas tanto de PSNR como de SSIM retornan valores de 0 y 1, indicando que la imagen mejorada es igual a la de referencia. Se consideran ciertas razones por las cuales se obtuvieron estos resultados: optimización por parte del hardware de Nvidia, carencia de ruido en todas las imágenes de resonancia magnética, disparidad con respecto al valor máximo de intensidad de color en las métricas PSNR/SSIM, propiedades de la imagen de referencia utilizada, principalmente.

De esas posibilidades, una de especial interés para experimentación, en trabajos futuros, es aplicar rangos dinámicos en la intensidad de color para las métricas cuantitativas.

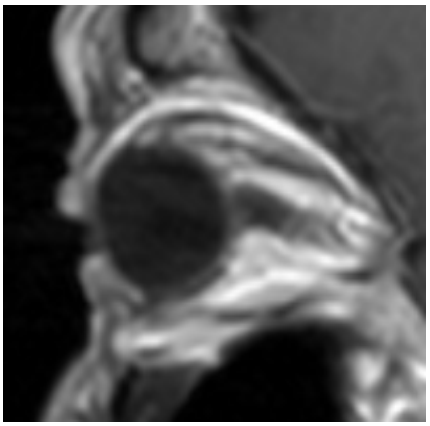
Imágenes mejoradas

En las Figuras 5.36a 5.39 se observa un acercamiento de las imágenes mejoradas mediante los tres algoritmos de súper resolución y la imagen de referencia correspondiente para apreciar de una mejor manera los detalles en sus texturas después del aumento, todo esto realizado en la GPU.

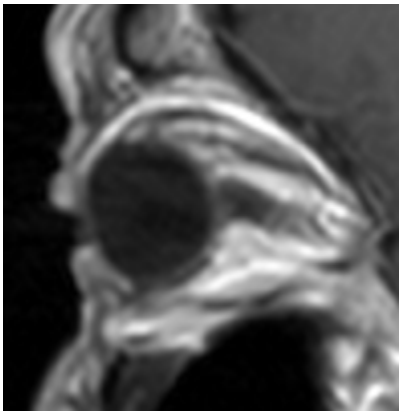
Los resultados de la implementación en GPU muestran diferencias en lo que se refiere a las texturas de las regiones de tejido blando y duro. La Figura 5.36 muestra las imágenes mejoradas por los algoritmos de súper resolución junto con la imagen de referencia.



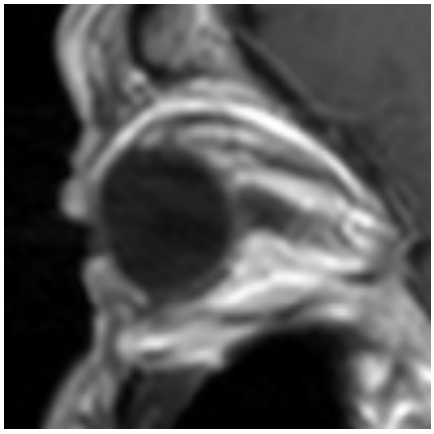
(a) NN.



(b) BL.



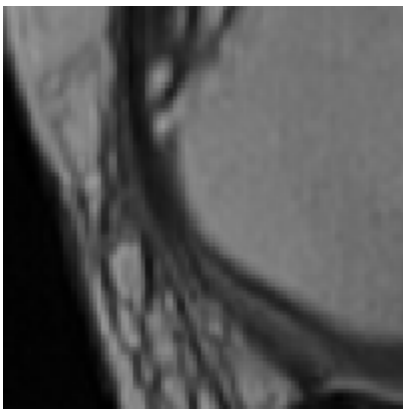
(c) BC.



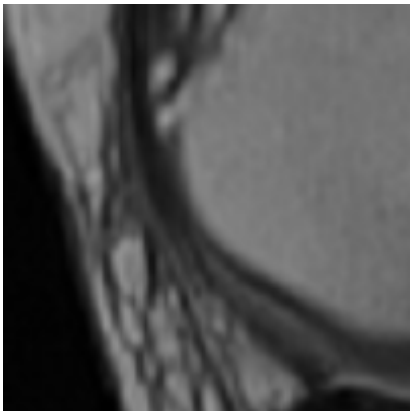
(d) Referencia.

Figura 5.36: Comparación de las imágenes en los estudios del cráneo.

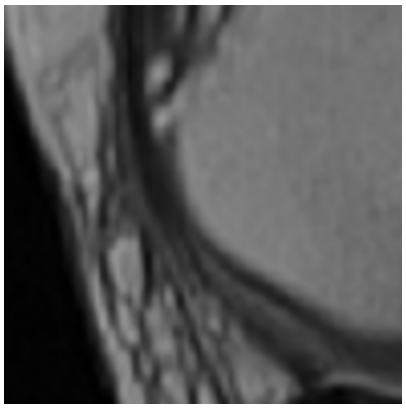
Para las imágenes de la rodilla, mostradas en la Figura 5.37, se identifica una mejora en lo que se refiere a la nítides de las regiones que hacen frontera en la zona oscura. Fuera de eso preserva las observaciones previas.



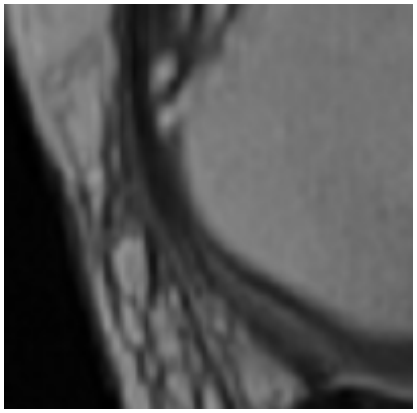
(a) NN.



(b) BL.



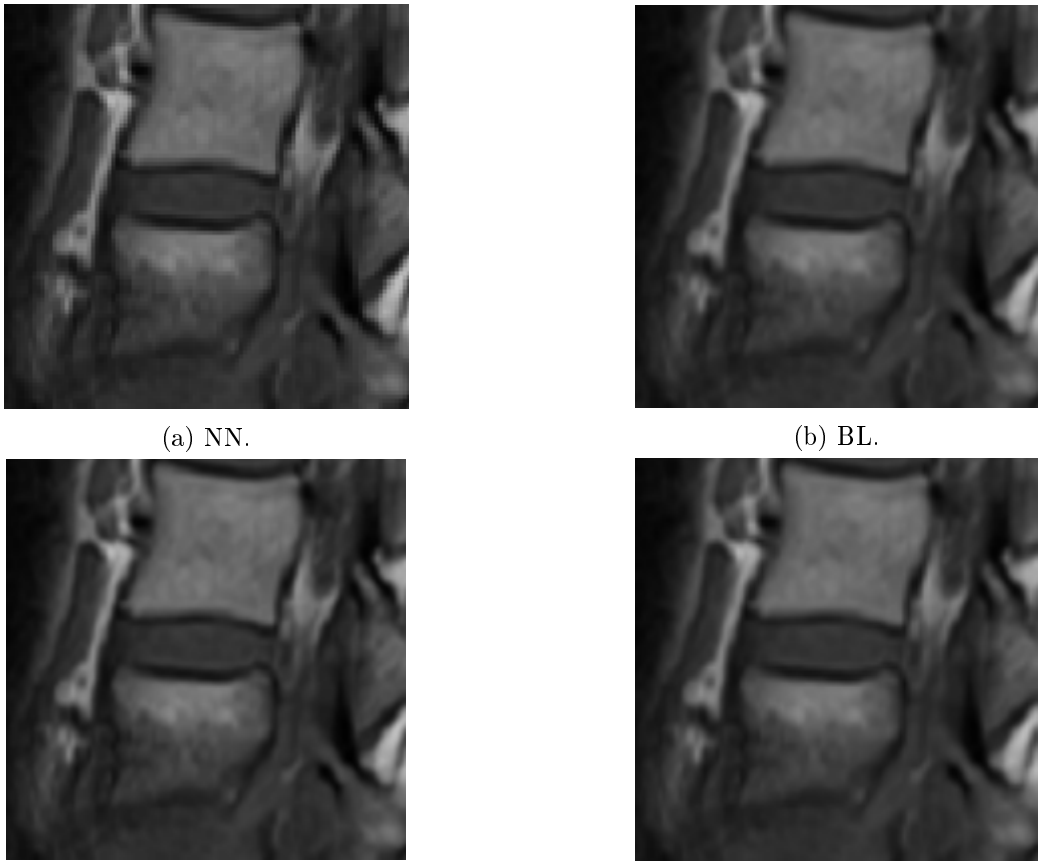
(c) BC.



(d) Referencia.

Figura 5.37: Comparación de las imágenes en los estudios de la rodilla.

Por último en los estudios de la columna (Figura 5.38) se observan también la mejora en las fronteras de región oscura, similar a las imágenes de la rodilla.

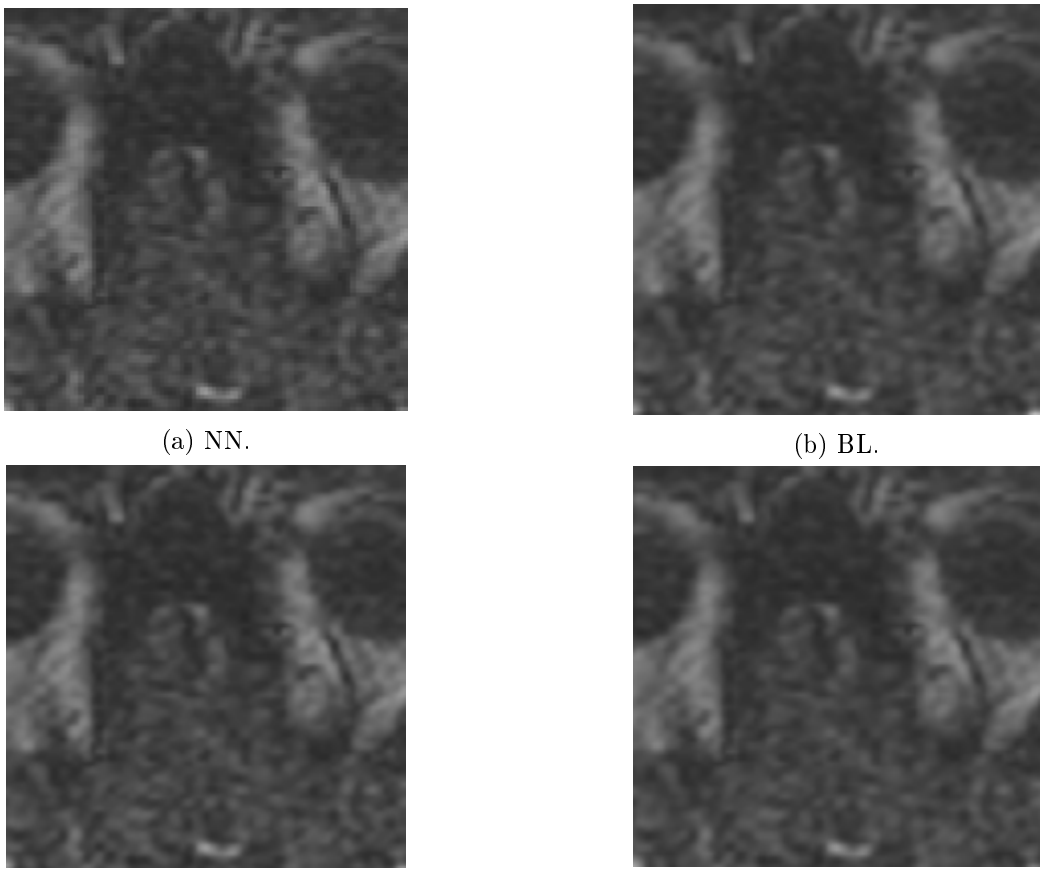


(a) NN. (b) BL.

(c) BC. (d) Referencia.

Figura 5.38: Comparación de las imágenes en los estudios de la columna.

En la base de datos pública, las imágenes Ryder de la Figura 5.39 muestran ciertas mejoras en las áreas oscuras, pero debido a la distorsión original de la imagen estás no son fácilmente visibles para el ojo humano. Por supuesto, las mejoras son difícilmente notables en las imágenes mejoradas por la interpolación bilineal (b) y bicúbica (c).



(a) NN. (b) BL.

(c) BC. (d) Referencia.

Figura 5.39: Comparación de las imágenes en los estudios de Ryder.

5.4.3. Discusión general de los resultados obtenidos

Los experimentos realizados en la CPU y GPU tienen un entorno de pruebas similar, siguiendo la misma metodología de experimentación. Los valores obtenidos de ambas implementaciones muestran las siguientes diferencias:

- Los valores en las métricas PSNR y SSIM son mayores en la implementación de GPU.
- El tiempo de procesamiento en las implementaciónes de GPU fue reducido por un factor de 5.8 con respecto a la implementación en CPU.
- Se alcanzó un valor optimizado para el algoritmo de interpolación bilineal en las métricas cuantitativas, siendo el sistema implementado en la GPU.

El valor optimizado en la implementación del GPU no fue esperado. Obtener una imagen que ha sido mejorada de manera exacta causa la necesidad de comprobar que los resultados obtenidos en ambas implementaciones sean precisos.

No es necesario asumir que los algoritmos tengan algún desperfecto al momento de ser escritos e implementados. Las causas se extienden desde los paramétros de las métricas de calidad hasta la configuración propia del GPU.

5.5. Análisis

Se presenta el análisis de los resultados obtenidos de la implementación del sistema exponiendo las observaciones de los resultados, la comprobación de estos mediante una búsqueda breve en la literatura y ciertas pruebas adicionales y la conclusión final.

5.5.1. Observaciones de los resultados

De los resultados obtenidos se obtuvieron las siguientes observaciones.

1. El método de interpolación bicúbica tarda el doble que el resto de los algoritmos.
2. El vecino más cercano es el más rápido en cuestión de tiempo.
3. El algoritmo de interpolación bilineal es el más efectivo con respecto a calidad de imagen.
4. En las implementaciones en GPU, la calidad de las imágenes llega a valores optimizados usando la interpolación bicúbica.
5. La velocidad de los algoritmos se incrementó por un factor de 5.88.

Las observaciones 1 y 2 están acorde a lo que se ha encontrado en la literatura. Sin embargo, la observación 3 presenta el caso de que la interpolación bilineal tiene valores superiores a la interpolación bicúbica en cuestión de mejora de imágenes, llegando a valores de 0 para PSNR, y 1 para SSIM. Esto contradice lo que está expuesto en la literatura sobre la interpolación bicúbica, al tener mejores propiedades de aumento debido a su método avanzado de interpolación no lineal. Para comprobar la veracidad de los resultados, se busco fundamentos de otros trabajos donde se haya implementado los algoritmos de interpolación bilineal y bicúbica.

La observación 4 muestra una optimización de la súper resolución por interpolación bilineal devolviendo imágenes con una resolución superior idénticas a las de referencia. Por ende corroborando lo encontrado en la literatura sobre GPU, los algoritmos de interpolación son optimizados debido al diseño de hardware en las GPU's modernas, dando mejores resultados de los esperados originalmente (mejorar el tiempo de procesamiento). Sin embargo, es necesario comprobar la observación 3 para justificar los resultados.

Por último, la observación 5 muestra que el tiempo de procesamiento de los algoritmos se aumentó por un factor de 5.88 en los tres que se seleccionaron. Cumpliendo con los resultados esperados del proyecto de tesis sobre reducir el tiempo de procesamiento mediante un sistema embebido de alto rendimiento.

5.6. Comprobación de resultados

Se busco en la literatura fundamentos sobre el uso de los métodos de súper resolución clásicos por medio de interpolación bilineal y bícubica. Lo que se encontró es conocimiento general sobre su uso, trabajos describiendo los resultados obtenidos bajo ciertos casos y finalmente, como resultado de está búsqueda, se realizo una evaluación con el sistema creado, usando imágenes estándar y comprobar la veracidad de los resultados.

5.6.1. Conocimiento general

La comunidad general de desarrolladores en aplicaciones de procesamiento de imágenes ha discutido el uso entre la interpolación bilineal y bicúbica en varias aplicaciones, enfocándose en las condiciones sobre las que éstas trabajan. Muchos de esos temas de interés son:

- Las tarjetas de vídeo usan de manera extensiva los algoritmos basados en interpolación.
- Debido a que la interpolación bicúbica utiliza técnicas complejas de procesamiento y no presenta diferencias significativas con la bilineal, ésta es usualmente descartada para el desarrollo de aplicaciones.
- La escala de los pixeles es un factor que afecta el proceso de mejora.
- En algunos casos, la interpolación bilineal tiene mejores resultados en encoger las imágenes mientras que la bicúbica es mejor para el aumento.
- La resolución de las imágenes llega a afectar los resultados.
- Se menciona mucho que la diferencia, entre los resultados obtenidos con interpolación bilineal y bicúbica, es mínima y varían con respecto a la implementación.
- Muchos usuarios consideran a la interpolación bilineal como un mejor algoritmo debido a su velocidad, antepuesto a la calidad de imagen.

La información que se ha recopilado en la discusión general es una discrepancia que existe con respecto a las implementaciones que se han realizado. Hay bastante factores que no se toman en cuenta o no se mencionan generalmente en la literatura por lo cual causa irregularidades en los resultados que se deseen replicar.

También se encontró que los distintos parámetros que existen en las métricas no tienen un estándar a seguir. Muchos de ellos son fijados por trabajos previos en la literatura o por el uso común que se le dedica a un área en específico. No hay un estándar definido como el ideal de manera objetivo. Además, el tipo de compresión que presentan las imágenes también afecta los resultados en las pruebas.

5.6.2. Otros trabajos en la literatura

Se buscaron más trabajos en la literatura para verificar los antecedentes de los métodos clásicos de súper resolución. Fueron encontrados los siguientes:

Mejora en la resolución de las imágenes mediante descomposición discreta y estacionaria de wavelets

En este trabajo (Demirel y Anbarjafari, 2011), los autores utilizaron como medidas de referencia para un nuevo algoritmo de SR a la interpolación bilineal y bicúbica. Usaron las siguientes imágenes estándar para realizar las pruebas:

- Lenna.
- Elaine.
- Baboon.
- Peppers.

Las cuales fueron aumentadas a una resolución de 128x128 a 512x512. Sin embargo no se hace referencia a más detalles con respecto a las imágenes utilizadas, tal y como lo es el tipo de formato de compresión (se observan que están a escala de grises pero no se menciona de manera explicita en el texto). Los resultados se muestran en la Tabla 5.9 , los cuales fueron evaluados mediante PSNR.

Tabla 5.9: Resultados en PSNR-Demirel.

Método	Lenna	Elaine	Baboon	Peppers
BL	26.34	25.38	20.51	25.16
BC	26.86	28.93	20.61	25.66

Conclusión: El algoritmo de interpolación bicúbica tiene mejor calidad en terminos de PSNR que el bilineal, pero, por una diferencia mínima. Además ciertos detalles con respecto a las propiedades de las imágenes fueron omitidos.

Comparación de sellado de imágenes usando un índice universal de calidad de imágenes

En este trabajo (H.S., H.L., y K.N., 2009), los autores utilizaron tres técnicas de súper resolución clásicas como medida de comparación para probar métricas de calidad basada en errores tales como PSNR.

Los detalles de la implementación se muestran de la siguiente forma:

- La implementación fue realizada en Matlab.
- Se utilizaron tres imágenes estándar (Cameraman, Clown, Soup Bubbles).
- La resolución de las imágenes era de 256 x 256.
- Formato tiff, jpeg y bmp.
- Se utilizaron los factores de aumento por 2, 4 y 8.

Aunque los resultados son bastante extensos,en síntesis, mejoran en función de que formato sea usado. La relación a este comportamiento se expresa como:

$$tiff < jpeg < bmp$$

Los resultados más altos de bmp para el algoritmo de interpolación bilineal y bicúbica fueron de **83.12** y **83.65**, a una escala de aumento por x2 pero sin especificación de la resolución de la imagen.

Conclusión: el aumento en la calidad de la imagen depende del formato a utilizar. A medida que se incrementas el factor de aumento, más difícil es mejorar las imágenes en cuestión de calidad. Por último, aún en los valores más altos la interpolación bicúbica supera a la bilineal, pero por muy poco.

Interpolación de imágenes que preserva la regularidad

En este trabajo (Carey, Chuang, y Hemami, 1999), los autores aportaron un nuevo método de interpolación mediante wavelets y lo compararon con algoritmos clásicos de interpolación bilineal y bicúbica. No proporciona detalles sobre la implementación o sus pruebas. Hace uso de la métrica PSNR para la medición de calidad y de las siguientes imágenes estándar:

- Lena.
- Couple.
- Mandrill.
- Peppers.
- Lake.
- LAX.

No provee más información acerca de las propiedades de las imágenes, los resultados se muestran en la Tabla 5.10:

Tabla 5.10: Resultados en PSNR-Carey

Imagen	**Bilinear**	**Bicúbica**
Lenna	27.3	27.5
Couple	26.8	27.0
Mandrill	23.4	23.6
Peppers	28.3	28.5
Lake	27.9	28.5
LAX	25.1	25.3

Conclusión: El trabajo no ofrece detalles sobre el entorno de desarrollo ni las propiedades de las imágenes. Los resultados muestran que el algoritmo de interpolación bicúbica es superior, en términos de PSNR, a la interpolación bilineal, pero por un margen muy pequeño.

Comparación de métodos de interpolación comunmente usados en imágenes

En este trabajo (Han, 2013), los autores hicieron un análisis de los tres algoritmos clásicos de súper resolución; Vecino más cercano, interpolación bilineal y bicúbica. Las pruebas fueron realizadas en Matlab 2009, utilizando la siguiente metodología.

Reduciendo una imagen a la mitad de su resolución, se agranda a su resolución original con los dos algoritmos de interpolación.

Las mediciones cuatitativas fueron realizadas con el error cuadrático de SNR (signal-to-noise ratio-razón señal a ruido) en 30 imágenes para las pruebas, de las cuales 10 fueron escogidas como representativas para esta sección del documento de tesis, siendo expuestos en la Tabla 5.11, donde el vecino más cercano, interpolación bilineal y bicúbica son expresados como NN, BL y BC respectivamente:

Tabla 5.11: Tabla de resultados en SNR-Han(2).

NN	BL	BC
19.1112	23.3023	23.4204
16.0059	18.4308	18.5899
20.0614	25.3736	25.6634
15.9497	18.7056	18.9392
17.1252	20.2387	20.9872
18.15564	22.137185	22.24938
14.40531	16.58772	16.73091
17.05219	21.56756	21.81389
12.75976	14.96448	15.15136
12.8439	15.179025	15.7404

Los resultados muestran que el algoritmo de interpolación bilinear tiene un desempeño más bajo que el de interpolación bicúbica, pero por una diferencia mínima.

No se muestran las propiedades de las imágenes, sin embargo mencionan una imagen de Lenna con una resolución original de 256x256 que fue diezmado a 64x64.

Conclusión: Este trabajo aporta una metodología de experimentación para probar los algoritmos clásicos de interpolación. Sin embargo, no menciona detalles del tipo de formato de las imágenes o de los parámetros de aumento.

Conclusión general

De los trabajos revisados previamente se hacen las siguientes observaciones:

- La interpolación bicúbica supera valores muy pequeños a la interpolación bilineal en cuestión de calidad.
- El formato de compresión en las imágenes afecta los resultados que se llegan a obtener mediante métricas de calidad.
- La mayoría de los trabajos no establece explícitamente el entorno de desarrollo con los cuales están trabajando o las propiedades de las imágenes.

A lo cual se llega a una importante observación: en la literatura las experimentaciones realizadas con algoritmos clásicos de súper resolución omiten para mención en sus documentos los factores en

los cuales realizan las experimentaciones y evaluaciones de las imágenes. No existe un estándar predefinido con el cual fijar los parámetros de experimentación. Por lo tanto, muchos de los trabajos realizados en este campo son hechos en entornos de desarrollo establecidos de manera arbitraria, según los autores.

5.6.3. Implementación con imágenes estándar

Dado lo mencionado en la literatura y a la falta de parámetros estándar a seguir, se propuso realizar pruebas en 12 imágenes estándar como método de evaluación del sistema y comprobar la veracidad de los resultados. Las imágenes utilizadas fueron:

- Lena.
- Jetplane.
- Blondewoman.
- Baboon.
- Pirate.
- Blackhairwoman.
- Cameraman.
- Peppers.
- Livingroom.
- House.
- Walkbridge.
- Lake

Los parámetros de las imágenes se establecen de la siguiente forma:

- Resolución de la imagen: 500x500 pixeles.
- Formato de compresión: Bmp.
- Profundidad de color: Escala de grises.

Los parámetros de las métricas PSNR y SSIM son los establecidos por la comunidad de desarrolladores de OpenCV (OpenCV_Community, 2017). Para los parámetros de súper resolución se estableció el de usar un factor de aumento x3 e incrementar las imágenes a una resolución de 1012x1012 pixeles.

Los resultados de los experimentos se observan en la Tabla 5.12:

Tabla 5.12: Resultados en las imágenes estándar en términos de Señal a Ruido de Pico y Similitud Estructural.

Bilinear	PSNR	SSIM
Lena	33.44	0.9371
Baboon	29.04	0.8871
Cameraman	35.77	0.9776
House	40.55	0.9853
Jetplane	35.27	0.9691
Pirate	34.71	0.9464
Peppers	36.45	0.9354
Walkbridge	31.47	0.9154
Blondew.	34.58	0.9383
Blackhairw.	42.12	0.9781
Livingroom	33.80	0.9390
Lake	33.44	0.9371
PROMEDIO	35.16	0.9460

Bicúbica	PSNR	SSIM
Lena	37.11	0.9609
Baboon	28.62	0.9136
Cameraman	36.97	0.9727
House	41.91	0.9808
Jetplane	36.22	0.9660
Pirate	35.01	0.9494
Peppers	36.43	0.9497
Walkbridge	31.57	0.9243
Blondew.	34.43	0.9488
Blackhairw.	42.83	0.9776
Livingroom	34.10	0.9436
Lake	33.96	0.9443
PROMEDIO	35.63	0.9511

Como evaluación adicional, se hicieron gráficas de líneas donde se representan los valores de PSNR y SSIM de las imágenes y la variación que tienen con respecto al uso de interpolación bilineal y bicúbica, como se puede apreciar en la Figuras 5.40 y 5.41:

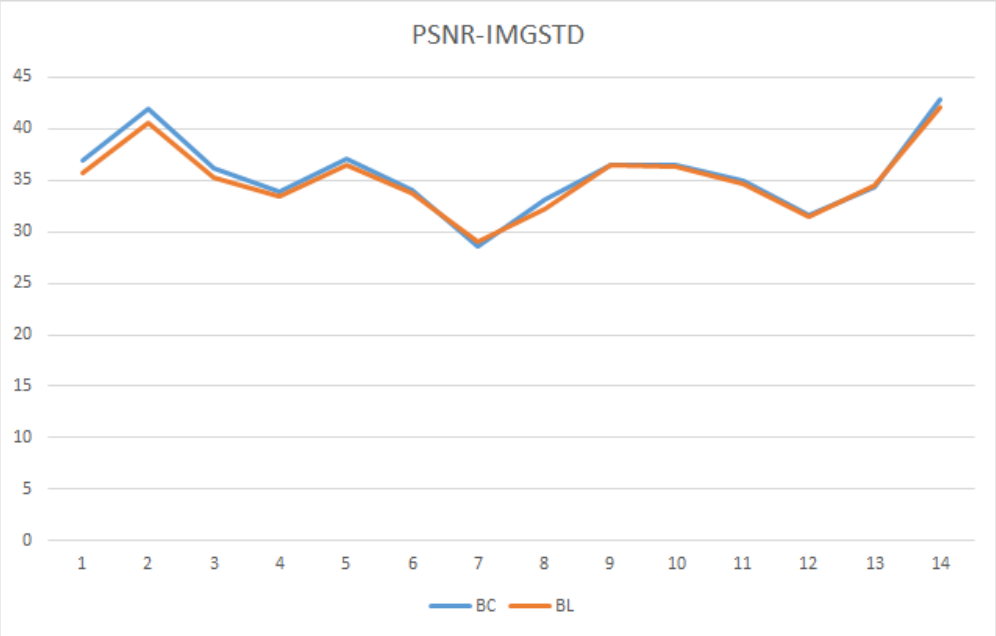


Figura 5.40: Valores de calidad en función de PSNR.

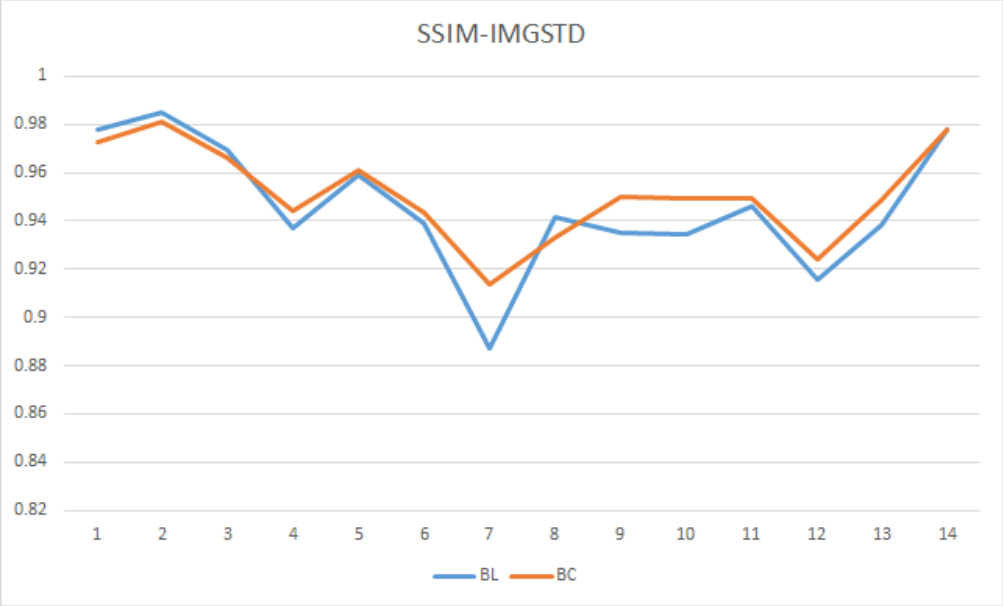


Figura 5.41: Valores de calidad en función de SSIM.

Discusión

De los resultados obtenidos se observan los siguientes rasgos a destacar.

1. Algunas imágenes dieron mejores resultados usando interpolación bilinear tanto para PSNR y SSIM.
2. Haciendo un promedio de los resultados, la interpolación bicúbica tiene mejor desempeño, aunque por un margen muy pequeño.
3. De la gráfica de líneas, se observa que la interpolación bicúbica tiene un rendimiento superior con ciertas imágenes en términos de PSNR, sin embargo, la interpolación bilineal se mantiene cercano a los valores por un margen mínimo.
4. También de la gráfica de líneas, en cuestión de SSIM, la interpolación bicúbica tiene mejores resultados con respecto a la bilinear, la cual sufrió de ciertos picos descendentes al momento de procesar ciertas imágenes, reflejando un desempeño más pobre que en PSNR.

Conclusión final

Dados los resultados obtenidos, los cuales muestran en valores promedios que la interpolación bicúbica tiene mejores resultados en la calidad de las imágenes que la bilineal, se comprueba que los algoritmos clásicos de súper resolución están acorde a lo expuesto en la literatura, por lo tanto el sistema de procesamiento de imágenes de resonancia magnética ha sido escrito correctamente y sus resultados son verídicos.

La observación 3 y 4 han sido comprobados y, por lo tanto, los resultados de la optimización del algoritmo de interpolación bilineal en la implementación de GPU son verídicos.

Como conclusión final: de la implementación de los tres métodos de súper resolución seleccionados en un sistema embebido de alto desempeño por GPU, haciendo evaluaciones de cientos de imágenes de fuentes tanto públicas como privadas, la interpolación bicúbica fue la que mostró mejores resultados en cuestión de métricas cuantitativas de calidad de imagen PSNR y SSIM, en un tiempo menor a la quinta parte a lo que se obtiene de implementaciones en CPU's convencionales.

Capítulo 6

Conclusiones

En este Capítulo se recapitula todo lo realizado en la tesis: objetivos completados, aportaciones que han sido presentadas en este proyecto, conclusión general de los resultados obtenidos, conclusión del trabajo y por último, los trabajos futuros.

6.1. Introducción

En este proyecto de tesis se realizó un sistema de procesamiento de imágenes que mejora la resolución de imágenes de resonancia magnética mediante el uso de métodos clásicos de súper resolución, siendo implementado en un sistema embebido de alto rendimiento.

Se realizó un estudio del estado del arte en lo que se refiere al procesamiento de imágenes, con el objetivo de investigar los más recientes avances en métodos de súper resolución. En este documento de tesis, se reportan los trabajos más recientes de súper resolución en medicina, de uso general, trabajos en CUDA, trabajos en las métricas evaluadoras seleccionadas y por último, trabajos en sistemas embebidos que han sido usados en aplicaciones de alto rendimiento como la visión por computadora. De este estudio se estableció:

- Las técnicas clásicas de súper resolución a utilizar.
- EL fundamento teoríco que apoya el uso de sistemas embebidos y tarjetas gráficas (GPU).
- Las métricas cuantitativas de calidad.

Es notable mencionar que también se obtuvo una perspectiva amplia sobre las metodologías empleadas para evaluar la calidad de las imágenes mejoradas, lo cual apoyó a plantear una nueva metodología de evaluación.

Se buscaron bases de datos de imágenes de resonancia magnética con las cuales llevar a cabo la experimentación y evaluación de las técnicas de súper resolución seleccionadas. Dichas bases de datos fueron obtenidas gracias a la aportación del Instituto Tlaxcalteca de Atención Especializada a la Salud (ITAES), quienes otorgaron tres estudios enteros sobre imágenes de resonancia magnética de pacientes recientes. Posteriormente se buscó otra base de datos que fuera pública para fundamentar aún más los resultados que fueran obtenidos. Siendo en total 2705 imágenes de trabajo.

Tres algoritmos clásicos de súper resolución fueron seleccionados para ser evaluados; el vecino más cercano, interpolación bilineal y bicúbica. Los algoritmos fueron implementados en una computadora convencional de trabajo usando C++ junto con la librería de acceso libre, OpenCV. Se realizó una evaluación usando las tres bases de datos privadas y la pública, procesando estudios enteros de entre 200 a 1500 imágenes para determinar cual era el algoritmo más eficaz. La evaluación constó de utilizar las métricas cuantitativas Señal a Ruido de Pico (PSNR), Similitud estructural(SSIM) y el tiempo de procesamiento de cada algoritmo. **Los resultados mostraron que la interpolación bilineal era el algoritmo con mejor desempeño en cuestión de calidad de imagen y segundo mejor en tiempo de procesamiento**.

El sistema embebido seleccionado fue **Jetson TK1** (*NVIDIA Embedded Systems*, 2014), de Nvidia, debido a su capacidad de alto rendimiento y soporte de la plataforma de cómputo paralelo CUDA. Fue adquirido desde el sitio Amazon (*Amazon / Nvidia JetsonTK1*, 2017) con un costo de, apróximadamente, 5.000 pesos mexicanos (incluyendo envio). Se realizó la implementanción del sistema de procesamiento de imágenes en una computadora de escritorio convencional, para su posterior traslado a la tarjeta Jetson.

Utilizando el soporte de tarjetas gráficas en OpenCV, se realizo una versión optimizada del sistema utilizando las capacidades de paralelismo proveídas por CUDA. Se utilizó la misma metodología al evaluar los tres algoritmos clásicos de súper resolución, cuyos resultados se mantuvieron sin alteración con respecto a los obtenidos de la implementación de CPU; la interpolación bilineal sigue siendo mejor en cuestión de calidad de imagen. También se obtuvieron mejoras significativas a lo que se refiere el tiempo de procesamiento.

6.2. Conclusiones generales

Se presentan las siguientes conclusiones generales por capítulo presentado en este documento de tesis:

1. En el Capítulo de Introducción se presenta el tema a tratar, la problemática, los objetivos y alcances, y el método de solución.

   Se da el contexto bajo el cual se propuso este proyecto de tesis en lo que se refiere al área de inteligencia artificial y la medicina para aportar soluciones a problemáticas actuales en lo que se refiere a la calidad de experiencia de los pacientes.

2. El Capítulo de Estado del arte se presenta los trabajos más recientes en cuestión de súper resolución, CUDA, la medición de calidad en imágenes digitales y sistemas embebidos

   El estado del arte aportó un panorama más amplio sobre las tendencias actuales en el área de procesamiento de imágenes. Las aplicaciones conjuntas de machine learning y súper resolución se han incrementado de manera considerable en los últimos diez años, brindando oportunidades de desarrollo tecnológico tanto en el sector computacional como médico.

3. En el Capítulo de Algoritmos de Súper Resolución y métricas de evaluación se presentaron tres algoritmos clásicos de súper resolución basados en interpolación y dos métricas de calidad de uso común en el procesamiento de imágenes.

Los tres algoritmos: vecino más cercano, interpolación bilineal y bicúbica, son extensamente mencionados en la literatura como comparadores para nuevos métodos de súper resolución emergentes. La implementación en el sistema embebido propuesto ha mostrado pocas complicaciones debido a su simplicidad. Un objetivo de interés para trabajos futuros sería implementar métodos alternos de SR basados en machine learning.

4. En el Capítulo Implementación en el sistema embebido Jetson TK1 se presentó el trabajo de implementación en la tarjeta embebida de alto rendimiento seleccionada.

   La implementación de los productos embebidos de Nnvidia resultó más difícil de lo esperado. La carencia de documentación apropiada para realizar configuraciones tanto básicas como avanzadas en la tarjeta extiende el tiempo de trabajo técnico y no provee de un aprendizaje adecuado sobre el uso de la tarjeta. Una ventaja a considerar al respecto es la oportunidad de escribir documentación técnica didáctica para reducir el tiempo invertido en tecnicismos. Además, la configuración de los hilos en el GPU es una tarea complicada debido a la complejidad del mapeo óptimo en los algoritmos que son diseñados para trabajar en paralelo; un trabajo futuro, aunque no para este proyecto, sería investigar modelos adecuados para el paralelismo en función de un dominio en específico.

5. En el Capítulo de Experimentación y análisis de resultados se presentaron las bases de datos a evaluar; una privada y una pública, la metodología de experimentación, el entorno de desarrollo, los resultados de las evaluaciones; donde se expuso lo obtenido en las implementaciones de CPU y GPU en cuestión de calidad de imagen y tiempo de procesamiento, análisis mediante observaciones en los resultados y por último la comprobación de resultados mediante una segunda búsqueda en la literatura y pruebas adicionales con imágenes estándar.

   Los resultados de las implementaciones mostraron variaciones en función de los estudios utilizados. Estas variantes son inducidas por parámetros presenten en las métricas cuantitativas. Otra variable a considerar es la imagen de referencia, cuyas propiedades dependen del algoritmo utilizado para mejorarla. Para mejorar los resultados en trabajos futuros, se sugiere proponer una metodología de experimentación alterna.

6.2.1. Conclusión final del trabajo

La optimización del algoritmo de interpolación bilineal en la implementación de GPU resultó ser válido dadas las observaciones en el capítulo de Análisis.

Por lo tanto, como conclusión final se tiene que:

La implementación del método de súper resolución por interpolación bilineal en un sistema embebido de alto rendimiento por GPU, ejecutado en cientos de imágenes de fuentes tanto públicas como privadas, devuelve mejores resultados en cuestión de calidad de imagen que los métodos del vecino más cercano e interpolación bicúbica, y se obtiene un tiempo menor a la quinta parte en contraste con las implementaciones en CPU.

6.3. Objetivos completados

Los objetivos completados durante el trayecto del trabajo son:

- **Evaluación de los algoritmos:** tres algoritmos clásicos de súper resolución fueron evaluados utilizando imágenes de MRI: el vecino más cercano, la interpolación bilineal e interpolación bicúbica.
- **Perspectivas de las imágenes:** se utilizaron cuatro estudios distintos de las imágenes obtenidas, las cuales abarcan las siguientes partes del cuerpo: cráneo, rodilla y columna vertebral.
- **Implementación en hardware dedicado:** se implementó el sistema de procesamiento de imágenes en la tarjeta de alto rendimiento, Jetson TK1 de Nvidia; aprovechando las capacidades de cómputo paralelo en la plataforma CUDA.

Por lo tanto, se cumple de manera satisfactoria con el objetivo general, que ha sido:

Implementar en un sistema de procesamiento de imágenes, algoritmos clásicos de súper resolución para la manipulación de imágenes de resonancia magnética en un sistema embebido de alto rendimiento.

6.4. Aportaciones

Las aportaciones de este trabajo son las siguientes:

- Una nueva metodología de evaluación de imágenes basado en el análisis estadístico de un conjunto de más de 1000 imágenes.
- Realización de un estudio de imágenes médicas basado en fuentes públicas y privadas.
- La implementación de un sistema de procesamiento de imágenes médicas en un sistema embebido de alto rendimiento.
- El uso del cómputo paralelo para la optimización del procesamiento de imágenes médicas en un sistema embebido.

Los productos obtenidos durante el desarrollo de este documento de tesis se presentan en el apéndice B.1.

6.5. Trabajos futuros

Para los trabajos futuros, quedaron pendientes las siguientes mejoras que darán continuación al trabajo:

1. Implementar algoritmos de segmentación de regiones de interés para las imágenes mejoradas.
2. Aplicar filtros de ruido que afectan el rendimiento de los algoritmos de interpolación (reducir la aparición de desenfoque/artifacts, ruido rissiano).
3. Utilizar técnicas de machine learning con el propósito de habilitar la característica de detección al sistema.
4. Utilizar rangos dinámicos en los niveles a profundidad de color en las imágenes y para la métrica PSNR en función del valor máximo de error acumulado.

Capítulo 7

Referencias

Adame, D., y Clemente, R. (2012). Análisis de la calidad diagnóstica de imágenes de tomografía computarizada procesadas con un filtro bilateral. *Imagen diagnóstica*, *3*(2), 50–55.

Alberto, J. (2011). Eliminación de ruido impulsivo en imágenes a color , utilizando interpolación con funciones de base radial Impulsive noise elimination in color images using interpolation with radial-basis functions. *Ingeniería*, *16*(1).

Amazon | Nvidia JetsonTK1. (2017). Descargado 2017-11-01, de `https://www.amazon.com.mx/NVIDIA-Jetson-TK1-Development-Kit/dp/B00L7AWOEC/ref=sr{_}1{_}1/131-2270222-6850365?ie=UTF8{&}qid=1509556479{&}sr=8-1{&}keywords=jetson+tk1`

Bhavsar, A. V., y Rajagopalan, A. N. (2012). Range map superresolution-inpainting, and reconstruction from sparse data. *Computer Vision and Image Understanding*, *116*(4), 572–591. Descargado de `http://dx.doi.org/10.1016/j.cviu.2011.12.005` doi: 10.1016/j.cviu.2011.12.005

Buell, D. (2011). *In Praise of An Introduction to Parallel Programming* (ELSEVIER, Ed.). Morgan Kaufmann. Descargado de `http://aims.me.cycu.edu.tw/courses/101-2/IPMC/lecture{_}material/IntroductiontoParallelProgramming-PeterPacheco(2010).pdf` doi: 10.1007/978-1-4471-2736-9

Campmany, V., Silva, S., Espinosa, A., Moure, J. C., Vázquez, D., y López, A. M. (2016). GPU-based pedestrian detection for autonomous driving. *Procedia Computer Science*, *80*, 2377–2381. doi: 10.1016/j.procs.2016.05.455

Carey, W. K., Chuang, D. B., y Hemami, S. S. (1999). Regularity-preserving image interpolation. *IEEE Transactions on Image Processing*, *8*(9), 1293–1297. doi: 10.1109/83.784441

Chac, A., y Espinosa, A. (2016). Embedded real-time stereo estimation via Semi-Global Matching on the GPU. *Procedia Computer Science*, *80*, 143–153. doi: 10.1016/j.procs.2016.05.305

Chouchene, M., Sayadi, F. E., Atri, M., y Tourki, R. (2013). Integral image computation on GPU. *2013 10th International Multi-Conference on Systems, Signals and Devices, SSD 2013*, 1–4. doi: 10.1109/SSD.2013.6564007

Cuadra, C., Alonso, F., Duran, D., y Fernando, J. (s.f.). Jetson hardware integrado.

Dawson, L., y Stewart, I. A. (2014). Accelerating ant colony optimization-based edge detection on the GPU using CUDA. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014* , 1736–1743. doi: 10.1109/CEC.2014.6900638

Demirel, H., y Anbarjafari, G. (2011). IMAGE resolution enhancement by using discrete and stationary wavelet decomposition. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, *20*(5), 1458–1460. doi: 10.1109/TIP.2010.2087767

Derenyi, E. E., y Saleh, R. K. (1989). *Effect Of Resampling On The Geometric And Radiometric Fidelity Of Digital Images* (Vol. 2). doi: 10.1109/IGARSS.1989.578861

Eklund, A., Dufort, P., Forsberg, D., y LaConte, S. M. (2013). Medical image processing on the GPU - Past, present and future. *Medical Image Analysis*, *17*(8), 1073–1094. Descargado de `http://dx.doi.org/10.1016/j.media.2013.05.008` doi: 10.1016/j.media.2013.05.008

Feng, C., Zhang, X., y Gao, Z. (2015). An Improved Image Super Resolution and Its Parallel ImplementationBased on CUDA. *The Tenth Interntional Conference on Digital Information Management*(Icdim), 182–187.

Gonzales, R., y Batchelor, B. (1978). *Digital Image Processing* (Vol. 24) (n.º 11). Prentice Hall. Descargado de `http://digital-library.theiet.org/content/journals/10.1049/ep.1978.0474` doi: 10.1049/ep.1978.0474

Graba, F., Loquin, K., Strauss, O., France, M. C., Graba, E. F., y Comby, F. (s.f.). Guaranteed Reconstruction for Image.

Han, D. (2013). Comparison of Commonly Used Image Interpolation Methods. *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)*(Iccsee), 1556–1559. Descargado de `http://www.atlantis-press.com/php/paper-details.php?id=4822` doi: 10.2991/iccsee.2013.391

Hefnawy, A. A. (2013). An efficient super-resolution approach for obtaining isotropic 3-D imaging using 2-D multi-slice MRI. *Egyptian Informatics Journal*, *14*(2), 117–123. Descargado de `http://dx.doi.org/10.1016/j.eij.2013.03.003` doi: 10.1016/j.eij.2013.03.003

H.S., P., H.L., S., y K.N., B. M. (2009). Image Scaling Comparison Using Universal Image Quality Index. *2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies*, 859–863. Descargado de `http://ieeexplore.ieee.org/document/5375929/` doi: 10.1109/ACT.2009.218

Hu, J. (2014). Real Time Super Resolution Reconstruction for Video Stream based on GPU. *International Conference on Orange Technologies, IEEE*(20125153025), 9–12.

Isaac, J. S., y Kulkarni, R. (2015). Super resolution techniques for medical image processing. *Technologies for Sustainable Development (ICTSD), 2015 International Conference on*, 1–6. doi: 10.1109/ICTSD.2015.7095900

Jetson TK1 - eLinux.org. (2017). Descargado 2017-09-18, de `http://elinux.org/Jetson{_}TK1`

Kato, T., Hino, H., y Murata, N. (2015). Multi-frame image super resolution based on sparse coding. *Neural Networks*, *66*, 64–78. Descargado de http://dx.doi.org/10.1016/j.neunet.2015.02.009 doi: 10.1016/j.neunet.2015.02.009

Malczewski, K. (2013). Super-resolution Magnetic Resonance Image Reconstruction with k-t SPARSE-SENSE at its Core. *Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), 2013*, 182–185.

Microdicom viewer. (s.f.). Descargado 2017-12-07, de http://www.microdicom.com/downloads.html

NVIDIA. (2017). CUDA C programming guide, v4.2. *Changes*(January).

NVIDIA Embedded Systems. (2014). Descargado 2017-11-01, de http://www.nvidia.com/object/jetson-tk1-embedded-dev-kit.html

NVIDIA GameWorks Documentation - Introduction. (2014). Descargado 2017-09-18, de http://docs.nvidia.com/gameworks/index.html{#}technologies/mobile/opencv{_}intro.htm

OpenCV_Community. (2017). *Video Input with OpenCV and similarity measurement.* Descargado 2017-09-12, de http://docs.opencv.org/2.4/doc/tutorials/highgui/video-input-psnr-ssim/video-input-psnr-ssim.html

Pappas, T. N., Safranek, R. J., y Hill, M. (1999). Percepual Criteria for Image Quality Evaluation. , *43*(908).

Parker, J. a., Kenyon, R. V., y Troxel, D. E. (1983). Comparison of interpolation methods for image resampling. *IEEE Transactions on Medical Imaging*, *2*(1), 31–39. Descargado de http://www.ncbi.nlm.nih.gov/pubmed/18230471 doi: 10.1109/42.7784

Piccialli, F., Cuomo, S., y De Michele, P. (2013). A regularized MRI image reconstruction based on hessian penalty term on CPU/GPU systems. *Procedia Computer Science*, *18*, 2643–2646. Descargado de http://dx.doi.org/10.1016/j.procs.2013.06.001 doi: 10.1016/j.procs.2013.06.001

Pooley, R. A. (2005). *Fundamental Physics of MR Imaging* (Vol. 25) (n.º 4). Radiological Society of North America. Descargado de http://pubs.rsna.org/doi/10.1148/rg.254055027 doi: 10.1148/rg.254055027

Ramos, G. G. J., y Garcia, J. C. S. (2015). Embedded system for real-time person detecting in infrared images/videos using super-resolution and Haar-like feature techniques. *2015 12th International Conference on Electrical Engineering, Computing Science and Automatic Control*. doi: 10.1109/ICEEE.2015.7357980

Rasti, P., Demirel, H., y Anbarjafari, G. (2013). Iterative back projection based image resolution enhancement. *Iranian Conference on Machine Vision and Image Processing*, 237–240.

Rasti, P., Lüsi, I., Demirel, H., Kiefer, R., y Anbarjafari, G. (2014). Wavelet Transform Based New Interpolation Technique for Satellite Image Resolution Enhancement. *IEEE International Conference on Aerospace Electronics and Remote Sensing Technology*, 185–188.

Salvador, J. (2017). *Example-Based Super Resolution Example-Based Super Resolution*. Elsevier.

Shah, S. (2014). Real-time Image Processing on Low Cost Embedded Computers. *Technical Report No. UCB/EECS-2014-117*, 1–29.

Sim, S., Kim, J.-h., Yuan, Z., Kang, S.-m., y Cho, J.-d. (2015). Robust-Rotation Recognition Based on Contour Matching Using CUDA in Automation System. *IEEE International Symposium on Consumer Electronics (ISCE) Robust-Rotation*, 0–1.

Su, M., Tan, J., Lin, C. Y., Ye, J., Wang, C. H., y Hung, C. L. (2015). Constructing a mobility and acceleration computing platform with NVIDIA Jetson TK1. *Proceedings - 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security and 2015 IEEE 12th International Conference on Embedded Software and Systems, H*, 1854–1858. doi: 10.1109/HPCC-CSS-ICESS .2015.212

Velasco, N. F., Rueda, A., Santa Marta, C., y Romero, E. (2017). A sparse Bayesian representation for super-resolution of cardiac MR images. *Magnetic Resonance Imaging*, *36*, 77–85. Descargado de `http://dx.doi.org/10.1016/j.mri.2016.10.009` doi: 10.1016/j.mri.2016.10.009

Wang, Z., Bovik, a. C., Sheikh, H. R., y Simmoncelli, E. P. (2004). Image quality assessment: form error visibility to structural similarity. *Image Processing, IEEE Transactions on*, *13*(4), 600–612.

Yue, L. (2016). Image super-resolution : The techniques , applications , and future. *Signal Processing*, *128*, 389–408. Descargado de `http://dx.doi.org/10.1016/j.sigpro.2016.05.002` doi: 10.1016/j.sigpro.2016.05.002

Zindros, D. (s.f.). *A Gentle Introduction to Algorithm Complexity Analysis*. Descargado 2017-12-14, de `https://discrete.gr/complexity/`

Appendices

Apéndice A

Bases de datos

A.1. BrainWeb: Simulated Brain Database. McGill University

http://brainweb.bic.mni.mcgill.ca/brainweb/

A.2. USF range image database. University of Southern Florida

http://marathon.csee.usf.edu/range/DataBase.html

A.3. Image and video quality assessment research at LIVE

http://live.ece.utexas.edu/research/quality/

A.4. Cancer imaging archive

http://www.cancerimagingarchive.net

Apéndice B

Productos

B.1. Publicación en la revista Pistas Educativas

En Octubre del año 2016, en la revista pistas educativas Num.120, fue publicado el artículo:

Pistas Educativas, No. 120, octubre 2016. México, Instituto Tecnológico de Celaya.

Súper resolución y mejora del algoritmo Canny para la detección de bordes en imágenes médicas

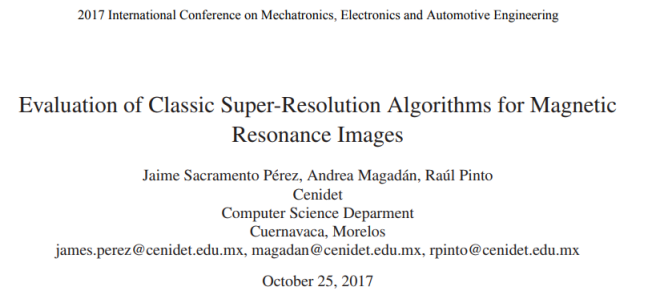
Jaime Sacramento Pérez Gutiérrez, Andrea Magadán Salazar, Raúl Pinto Elías, Manuel Mejía Lavalle

Centro Nacional de Investigación y Desarrollo Tecnológico, Interior Internado Palmira s/n, Col. Palmira, C.P. 62490, Cuernavaca, Morelos, Tel: (961) 203 9560

james.perez@cenidet.edu.mx, magadan@cenidet.edu.mx, rpinto@cenidet.edu.mx, mlavalle@cenidet.edu.mx

B.2. Publicación en el congreso de ICMEAE 2017

En Diciembre 28 del año 2017, en la revista computer society de la IEEE ICMEAE 2017, fue publicado el artículo:



2017 International Conference on Mechatronics, Electronics and Automotive Engineering

Evaluation of Classic Super-Resolution Algorithms for Magnetic Resonance Images

Jaime Sacramento Pérez, Andrea Magadán, Raúl Pinto
Cenidet
Computer Science Deparment
Cuernavaca, Morelos
james.perez@cenidet.edu.mx, magadan@cenidet.edu.mx, rpinto@cenidet.edu.mx

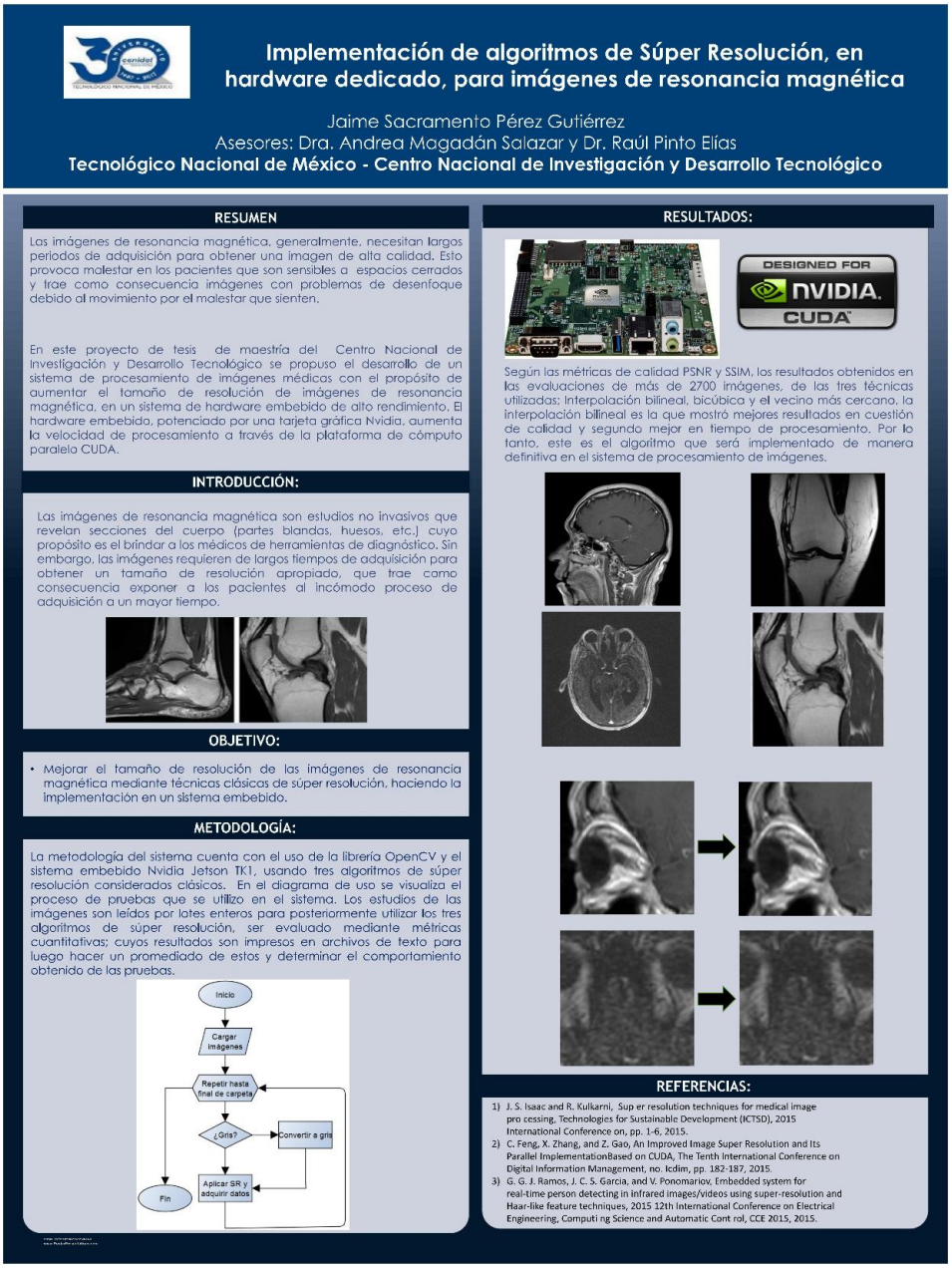
October 25, 2017

B.3. Estancia académica

Durante el mes de octubre se realizó una estancia académica en la Benemérita Universidad Autónoma de Puebla, donde se participó en la escuela de otoño en sistemas distribuidos en el curso de:**Introducción a la Arquitectura y Plataforma de Programación de Cómputo Paralelo CUDA.** impartido por el Instituto Nacional de Astrofísica, Óptica y Electrónica.

B.4. Exposición del proyecto en ICMEAE 2017

Durante el mes de Noviembre 2017 se expuso el proyecto de tesis en el congreso internacional de ingeniería mecatrónica, electrónica y automatización.



cenidet®

Centro Nacional de Investigación
y Desarrollo Tecnológico