CONAHCYT

CONSEJO NACIONAL DE HUMANIDADES
CIENCIAS Y TECNOLOGÍAS

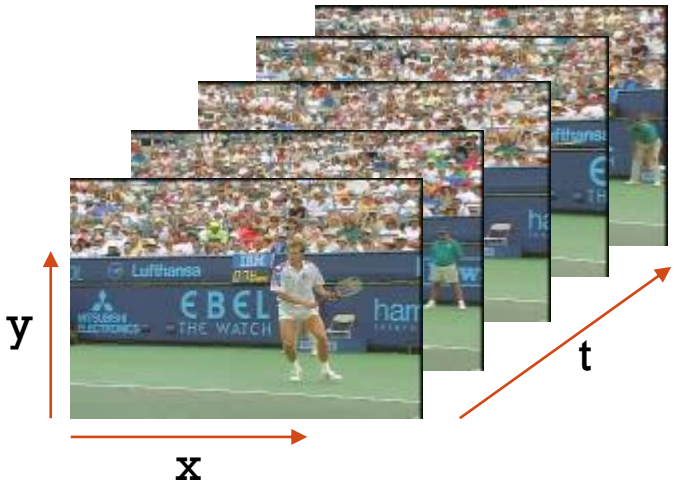
CIMAT
UNIDAD MÉRIDA

# PROYECTOS USANDO CÓMPUTO PARALELO

Dr. Francisco Javier Hernández López
CONAHCYT – CIMAT-Mérida
Tel.: +52 (999) 6885327, Ext.: 1306
fcoj23@cimat.mx, www.cimat.mx/~fcoj23

Ago-Dic 2024

# PROCESAMIENTO DE VIDEO




Desarrollar Algoritmos

- 0 o 1
- Imágenes Binarias
- Segmentación
- Posiciones de objetos a seguir
- Panoramas
- Detección y Rec. de Objetos, etc.

# DETECCIÓN DE CAMBIOS

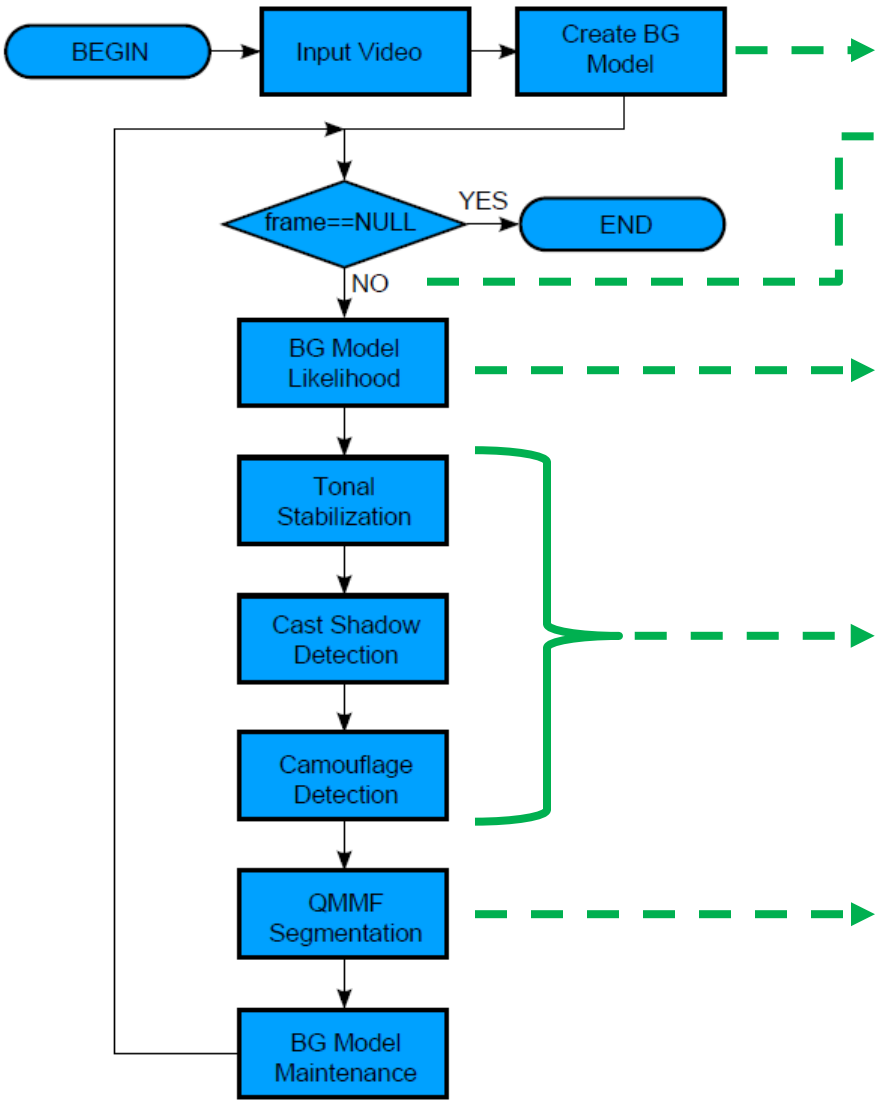
- El camino más simple para detectar cambios es:
  - Asumir que tenemos una imagen del fondo en la cual no existen objetos moviéndose

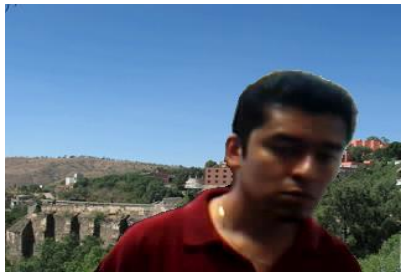
- Background (BG) → Pixeles que no cambian
- Foreground (FG) → Pixeles que han cambiado

- Pasos principales para la detección de cambios:
  a) Modelo del BG
  b) Inicialización del BG
  c) Actualización del modelo del BG
  d) Detección del FG

http://changedetection.net/
Bouwmans, T., Porikli, F., Höferlin, B., & Vacavant, A. *Background modeling and foreground detection for video surveillance. 2015*

# DET. DE CAMBIOS A PARTIR DE SEGM. PROB. (CDPS)



"Cambiar el Fondo"

Hernandez-Lopez, F.J. and Rivera, M., "Change Detection by Probabilistic Segmentation from Monocular View," Machine Vision and Applications, pages 1-21, 2013.

# COMPARACIÓN EN TIEMPO (MS) DE LA IMPLEMENTACIÓN

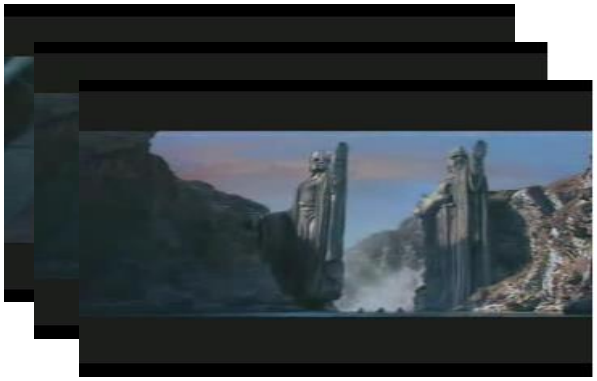
- Video de $640 \times 480$ pixeles

| Procedimiento | Tiempo (ms) | | Ganancia CPU/(CPU&GPU) |
|---|---|---|---|
| | CPU | CPU&GPU | |
| Cargar 5 modelos | 8.68 | **8.24** | 1.05 |
| Leer frame | **1.16** | 1.99 | 0.58 |
| Calcular verosimilitud | 13.14 | **0.36** | **36.5** |
| Estabilización tonal | 11.2 | **10.22** | 1.09 |
| Detección de sombras | 42.99 | **1.03** | **41.73** |
| Detección de camuflaje | 5.98 | **0.99** | 6.04 |
| Segmentación QMMF | 108.65 | **4.58** | **23.72** |
| Actualizar modelos | 26.52 | **5.01** | 5.29 |
| Desplegar resultados | 8.14 | 8.87 | 0.91 |
| **frames por segundo (fps)** | 4.59 | **30.26** | |

# AVSCREEN

- Herramienta de edición de video que modifica una región de un video por otro video o imagen en tiempo-real

I/V original

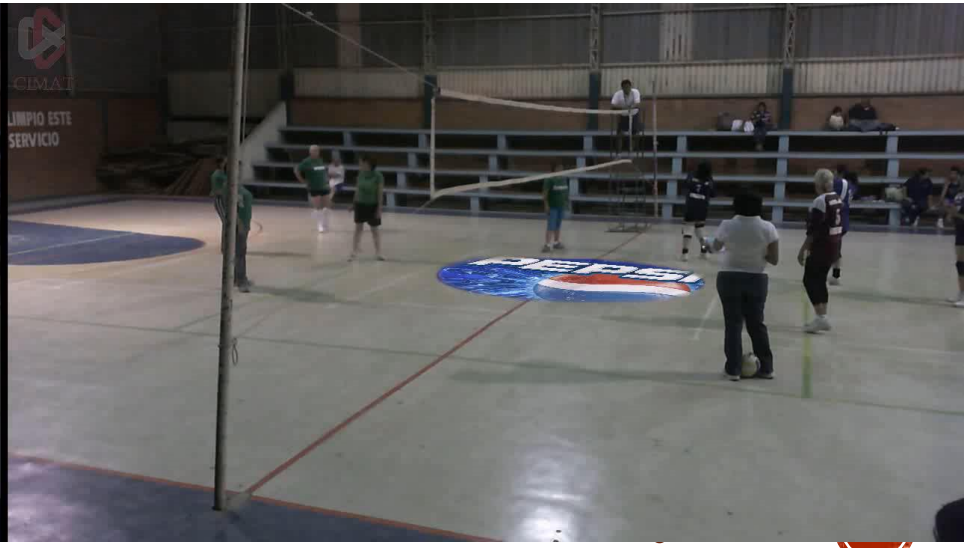


I/V nuevo

I/V editado

Hernandez-Lopez, F.J. and Rivera, M., "AVScreen: a Real-Time video augmentation method," J. of Real-Time Image Process., pages 1-13, 2013.

# AVSCREEN USANDO CÁMARAS ESTÁTICAS

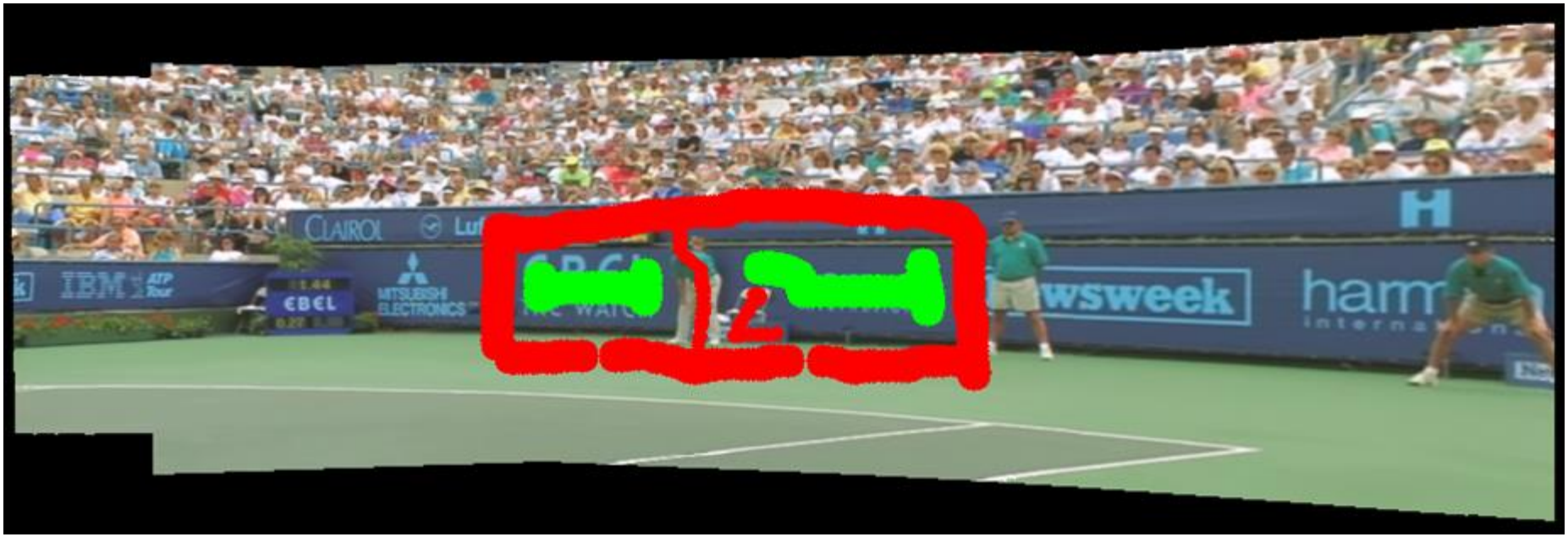




Video Original

Video Modificado

# AVSCREEN USANDO CÁMARAS INESTABLES

# AVSCREEN USANDO EL PANORAMA (INTERACCIÓN CON EL USUARIO)



# AVSCREEN USANDO EL PANORAMA (EDICIÓN DE VIDEO)

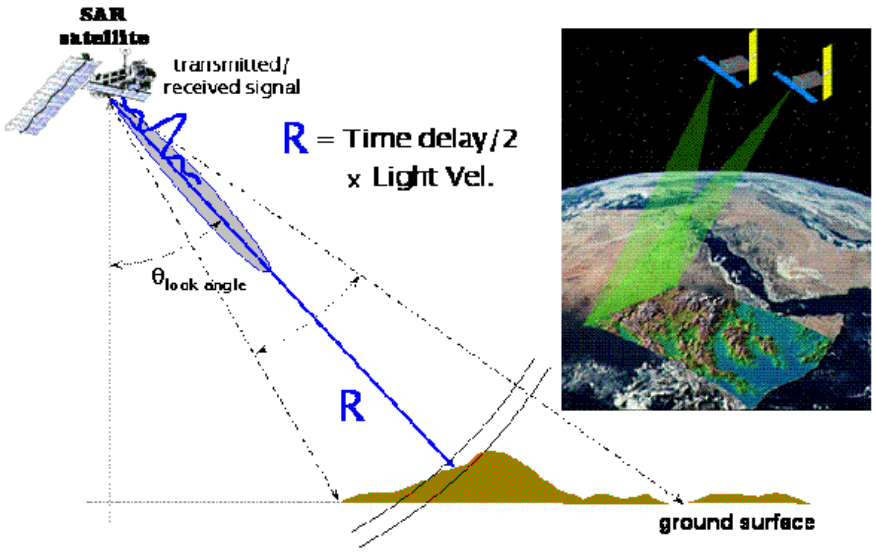

Video original


Video editado

# AVSCREEN USANDO EL PANORAMA

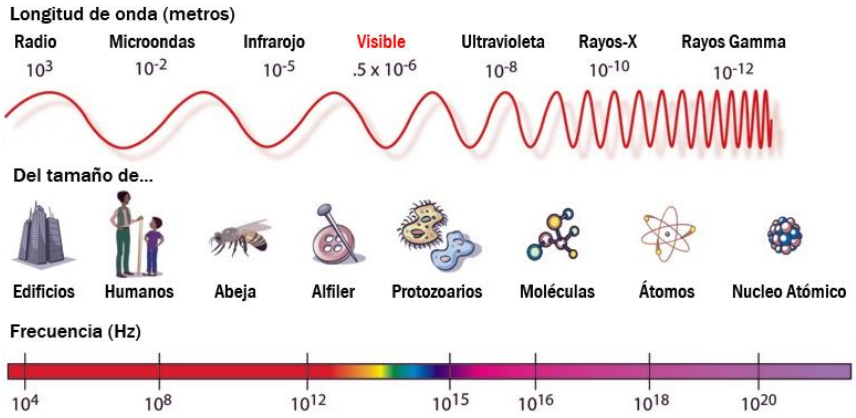
# AVSCREEN USANDO EL PANORAMA


# IMÁGENES SATELITALES




http://labs.cas.usf.edu/geodesy/sar.html

El radar de apertura sintética (SAR), es un radar activo que emite y recibe señales de microondas (1cm-100cm, 1GHz-300GHz).




En una imagen obtenida por radar un pixel contiene información de la amplitud y fase de un número complejo. A este formato se le conoce como SLC (Single-look-complex).

La fase de la señal $\phi(\vec{x})$ en cada pixel $\vec{x}$, queda envuelta en un intervalo de $(-\pi, \pi]$ o $(0, 2\pi]$ por el operador $\mathrm{atan2}(\,)$.

# DESENVOLVIMIENTO DE FASE

$$\phi_w(\vec{x}) = W\big(\phi_u(\vec{x})\big) = \phi_u(\vec{x}) + 2\pi\kappa(\vec{x})$$

$\phi_u(\vec{x})$ → la fase desenvuelta

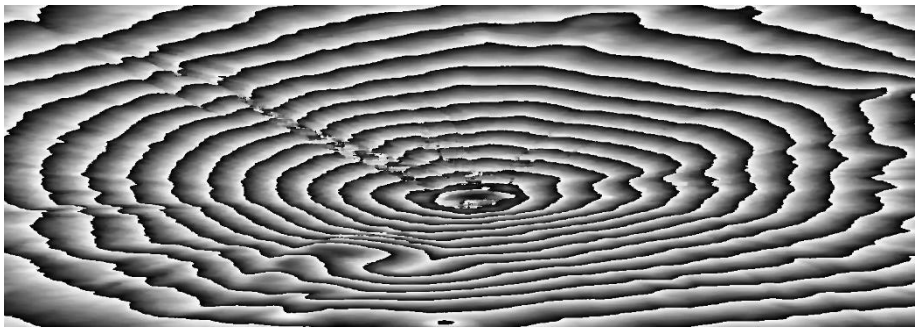
$\phi_w(\vec{x})$ → la fase envuelta

$\kappa(\vec{x})$ → valor entero en cada pixel $\vec{x}$

$W(\phi_u(\vec{x})) \in (-\pi, \pi]$ → Operador de envolvimiento

$$W(\phi_u(\vec{x})) = atan2\left(\frac{sen\big(\phi_u(\vec{x})\big)}{cos(\phi_u(\vec{x}))}\right)$$

A partir de una fase envuelta $\phi_w$, queremos recuperar la fase desenvuelta $\phi_u$



$\phi_w$

$\phi_u$

# IMPLEMENTACIÓN EN PARALELO DEL ARM


Multicore CPU

**Servidor K20**
Intel Xeon CPU E5-2620 v2 2.10 GHz
Ubuntu 14.04 (64 bits) 256 GB RAM
24 cores con hyperthreading

**Servidor K40**
Intel Xeon CPU E5-2690 v2 3.00 GHz
Ubuntu 14.04 (64 bits) 256 GB RAM
20 cores fisicos


XPC

**XPC 3120A**
1.1 GHz
57 cores
6GB RAM

GPU

**GPU Tesla K20**
0.71 GHz
13 SM, 2496 cores
5GB RAM

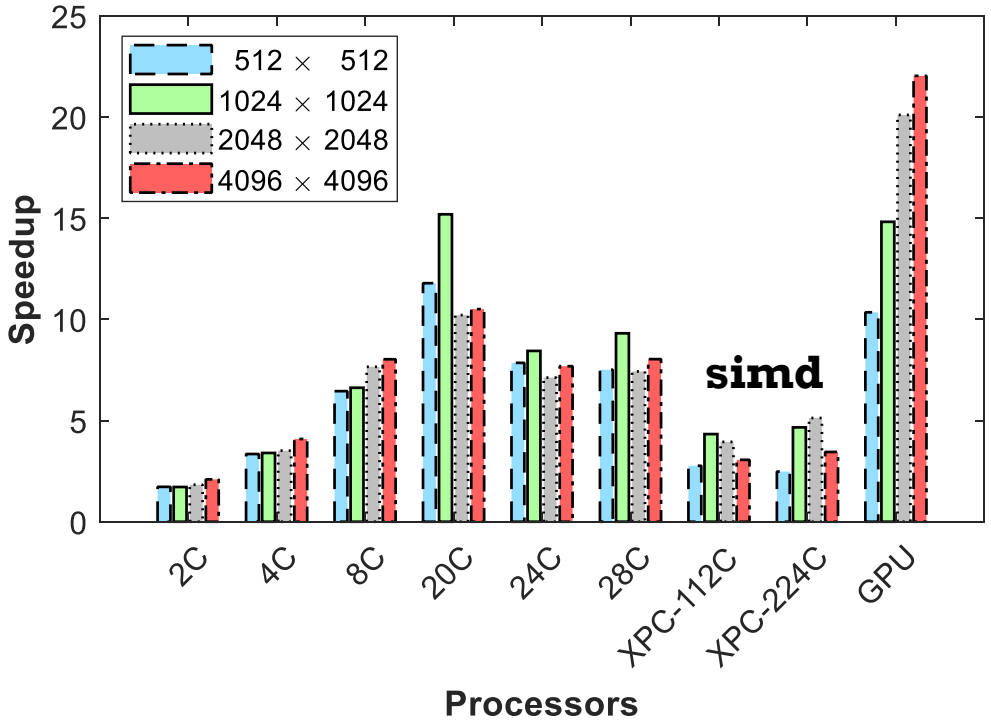
**GPU Tesla K40**
0.75 GHz
15 SM, 2880 cores
12GB RAM

Hernandez-Lopez, F. J., Rivera, M., Salazar-Garibay A., & Legarda-Saenz R. (2018). Comparison of multi-hardware parallel implementations for a phase unwrapping algorithm. Optical Engineering. ISSN: 0091-3286.

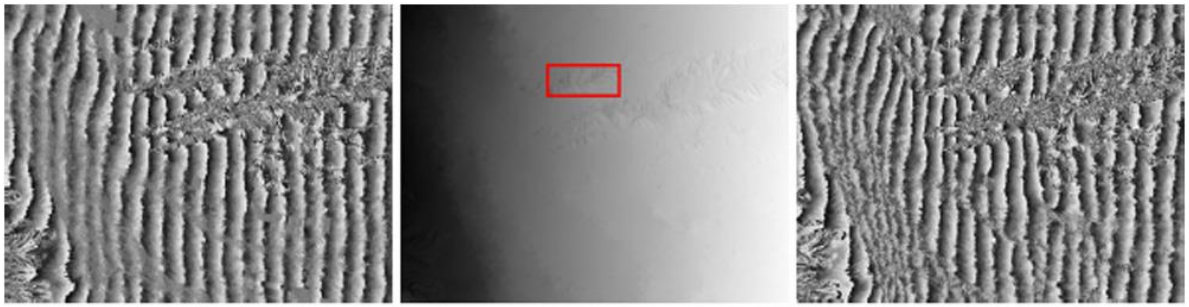
# OP_2, SERVIDOR_K40

Tiempo de procesamiento en segundos

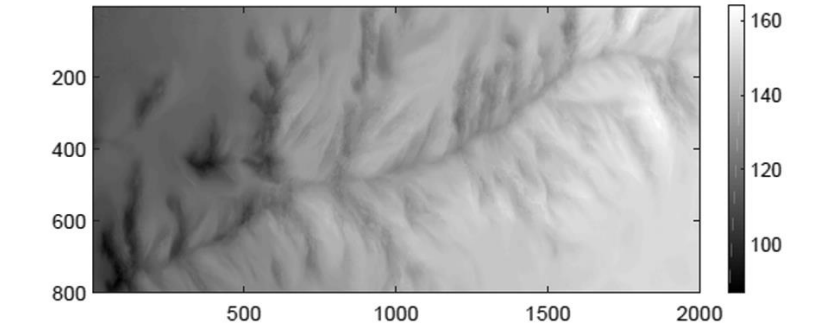
| | | Multicore CPU | | | | | | XPC | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Size of image | Serial 1*C* | 2*C* | 4*C* | 8*C* | 20*C* | 24*C* | 28*C* | 112*C* | 224*C* | GPU |
| $512^2$ | 26.3 | 15.3 | 7.9 | 4.1 | **2.2** | 3.4 | 3.5 | 9.5 | 10.6 | 2.5 |
| $1024^2$ | 107.2 | 62.4 | 31.5 | 16.2 | **7.1** | 12.7 | 11.5 | 24.8 | 23.0 | 7.2 |
| $2048^2$ | 515.7 | 281.9 | 146.7 | 67.4 | 50.5 | 72.4 | 69.4 | 130.7 | 100.6 | **25.7** |
| $4096^2$ | 2193.8 | 1047.1 | 536.6 | 273.2 | 208.8 | 285.4 | 273.0 | 716.9 | 636.2 | **99.6** |

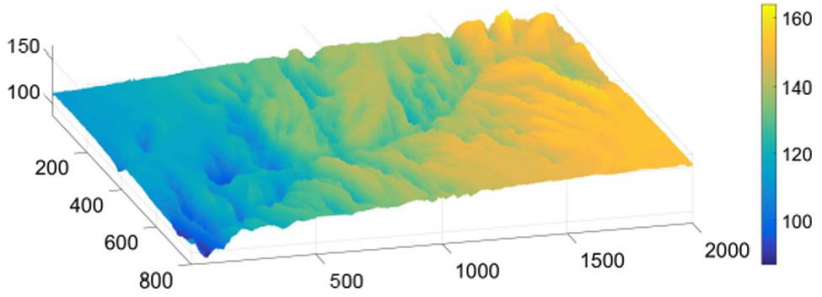


# DESENVOLVIENDO INTERFEROGRAMAS SAR



$\phi_w$ $\phi_u$ $W(\phi_u)$




$8000 \times 10500$ **pixeles**

**Parámetros:**

$\lambda = 10$

$\mu = 100$

$N = 7$ **niveles**

$K = 1000$ **iter.**

**Tiempo de Proc.:**

$Serial = 6216.8s$

$CPU_{20c} = 739.7s \sim 8x$

$GPU = 397.4s \sim 16x$

$XPC =$ **Memoria Insuficiente**

Fase envuelta tomada de un área de Phoenix, Arizona a partir de 2 imágenes SLC RADARSAT-2 usando SNAP. Las imágenes fueron adquiridas el 04 y 28 de Mayo del 2008.

# PARALELIZACIÓN MEDIANTE GPU DE LA ECUACIÓN DE POISSON 3D EN GEOMETRÍAS ARBITRARIAS

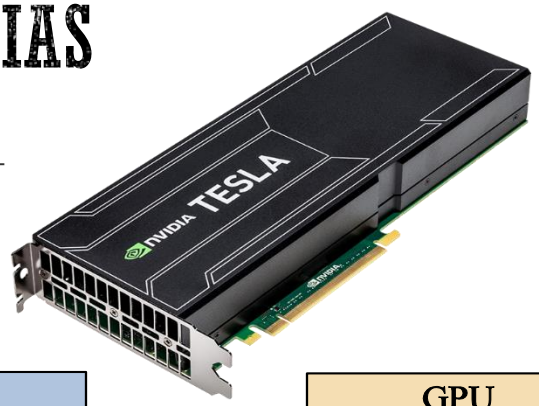
**Algorithm 1.** CUDA DTM version for the Jacobi parallel implementation

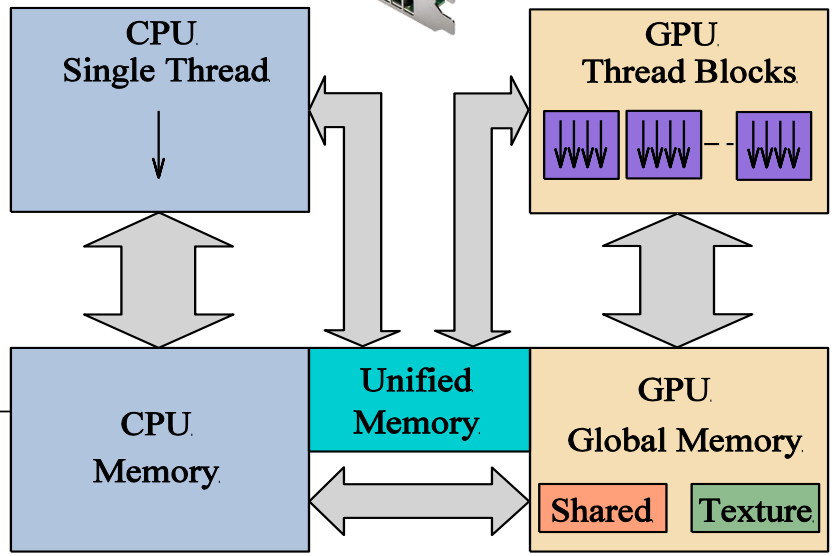
```
      use cudafor
      real*8,dimension(:),allocatable                    ::h_phi real *
      8,dimension(:),device,target,allocatable ::d_phi_real*8,
      dimension(:),device,target,allocatable ::d_phiNew_real*8,te
      xture,pointer                                      ::t_phi(:)
      real*8,dimension(:,:),managed,allocatable          ::m_error

      d_phi = h_phi   ! Copy memory from CPU to GPUt p
      hi =>d_phi ! Texture points to the device

500   continue
      error = 0.0d0
      call kJacobiMethod <<<grid,tBlock,shared_mem >>>()
      call kBoundaryCondition <<<grid,tBlock >>>()
      call k_Interpolation <<<gridv,tBlock >>>() d_p
      hi = d_phiNew
      error = sum(m_error)
      if (error.gt.1e-6)  goto 500

      h_phi = d_phi   ! Copy memory from GPU to CPU
```

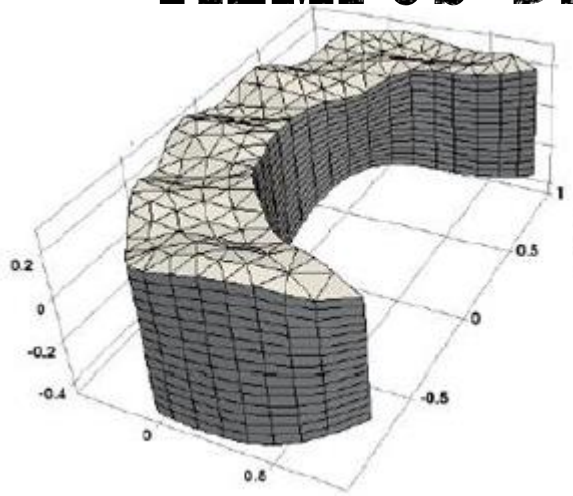


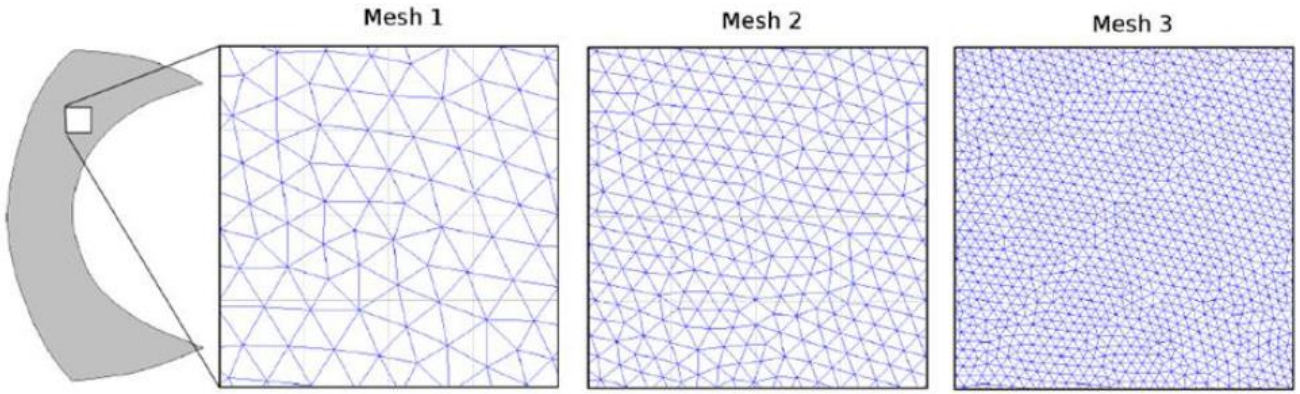




Para no migrar código a C/C++, se utilizó el compilador PGI CUDA Fortran

Uh Zapata, M., & Hernández-López, F. (2018). A GPU Parallel Finite Volume Method for a 3D Poisson Equation on Arbitrary Geometries. International Journal of Combinatorial Optimization Problems and Informatics, 9(1), pp. 3-11. ISSN: 20071558.

# TIEMPOS DE PROCESAMIENTO




Memoria: D → Device, T → Texture y M → Manage

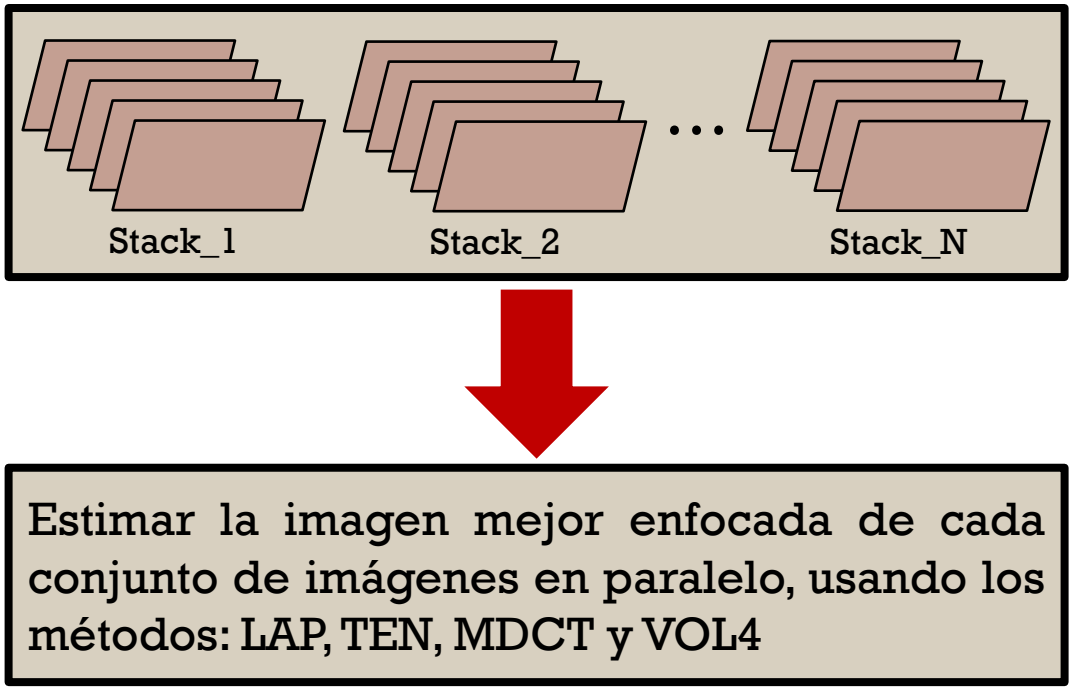
| Mesh | $Nz$ | Sequential Time (s) | CUDA D Time (s) | CUDA D Speedup | CUDA DT Time (s) | CUDA DT Speedup | CUDA DTM Time (s) | CUDA DTM Speedup |
|---|---|---|---|---|---|---|---|---|
| 1 | 32 | 4.52 | 0.60 | 7.53 × | 0.54 | 8.37 × | 0.53 | 8.53 × |
| 1 | 64 | 27.33 | 3.56 | 7.67 × | 3.21 | 8.51 × | 3.19 | 8.57 × |
| 1 | 128 | 193.37 | 35.42 | 5.45 × | 30.75 | 6.29 × | 30.14 | 6.42 × |
| 2 | 32 | 743.19 | 41.83 | 17.76 × | 26.50 | 28.04 × | 25.03 | 29.69 × |
| 2 | 64 | 1,738.82 | 91.78 | 18.94 × | 55.30 | 31.44 × | 53.82 | 32.31 × |
| 2 | 128 | 5,018.89 | 277.83 | 18.06 × | 163.56 | 30.69 × | 161.47 | 31.08 × |
| 3 | 32 | 11,734.84 | 626.34 | 18.74 × | 384.01 | 30.56 × | 364.93 | 32.16 × |
| 3 | 64 | 25,511.24 | 1276.06 | 20.00 × | 756.57 | 33.72 × | 737.09 | 34.61 × |
| 3 | 128 | 60,135.84 | 2934.38 | 20.49 × | 1708.39 | 35.20 × | 1681.11 | 35.77 × |

16.70 hrs.

0.47 hrs.

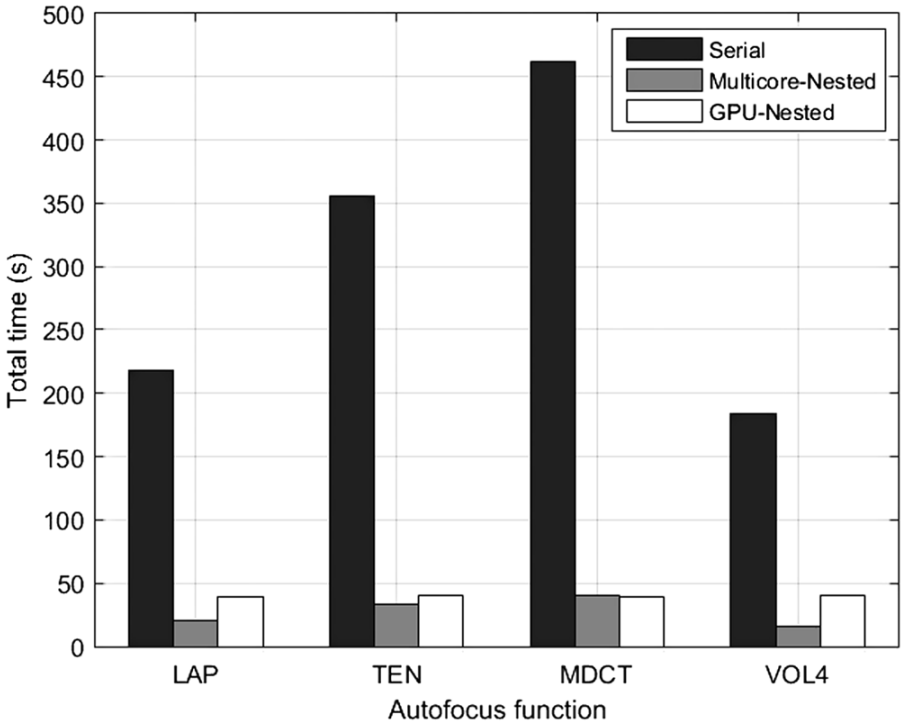
# IMPLEMENTACIONES PARALELAS PARA ACELERAR EL PROCESO DE AUTOENFOQUE EN IMÁGENES DE MICROSCOPIA

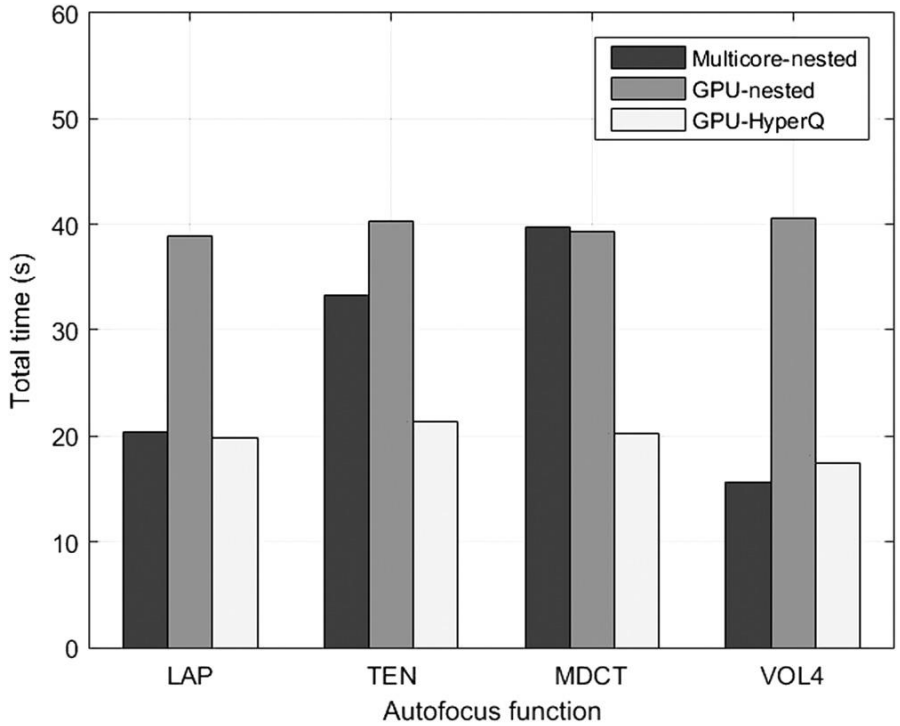
Microscopio
NIKON ECLIPSE 50i FLUORESCENCE




Juan C. Valdiviezo-N, Francisco J. Hernandez-Lopez, Carina Toxqui-Quitl, "Parallel implementations to accelerate the autofocus process in microscopy applications," J. Med. Imag. 7(1), 014001 (2020), doi: 10.1117/1.JMI.7.1.014001. ISSN print: 2329-4302, ISSN electronic: 2329-4310. Demo.

# TIEMPO DE PROCESAMIENTO

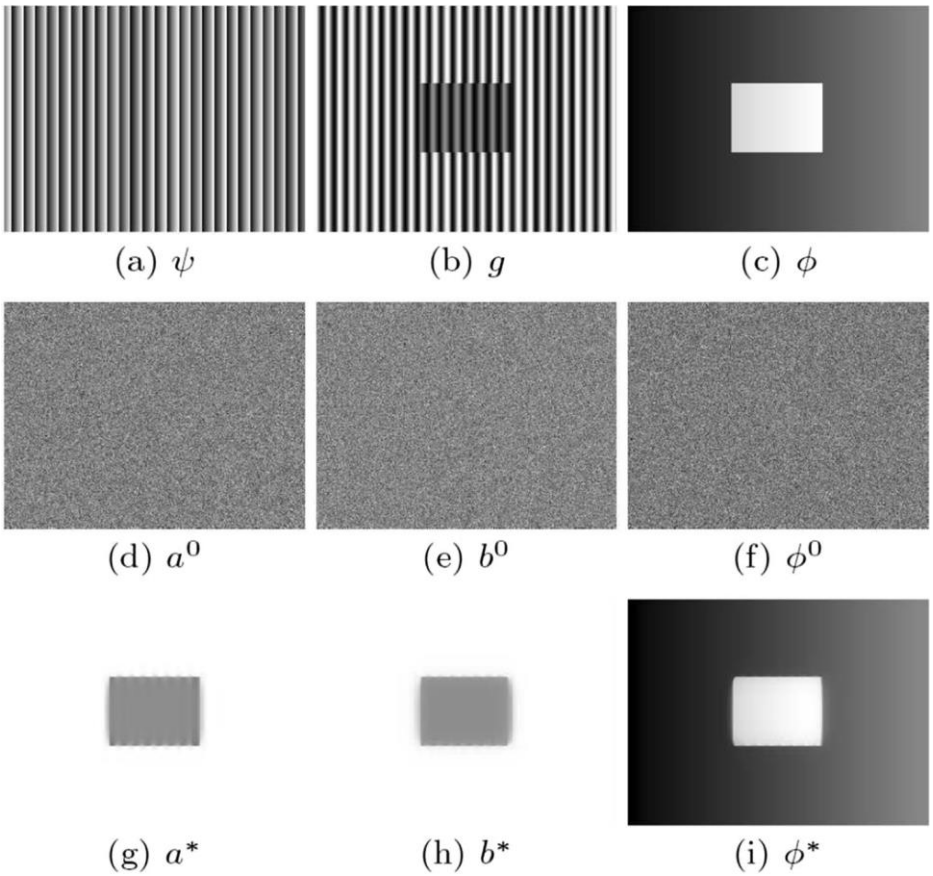



Tiempo total, procesando una base de datos de imágenes de Tuberculosis Micobacteriana, que consiste en **300** stacks de **20** imágenes (1200x1600 pixeles) en cada stack.

# IMPLEMENTACIÓN EN PARALELO DE UN ALGORITMO DE DEMODULACIÓN DE FASE

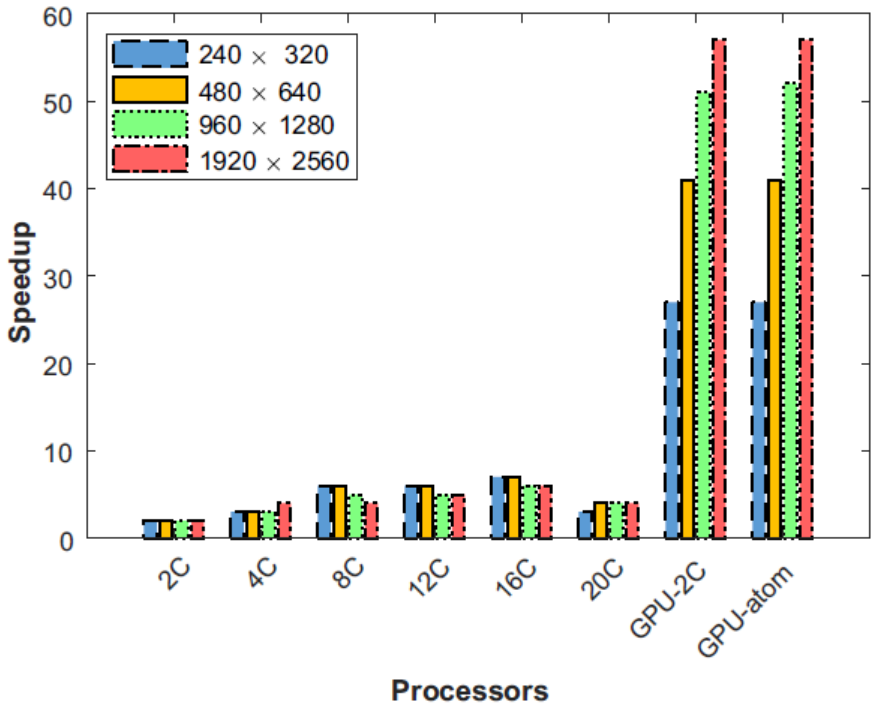
$$I(x,y) = a(x,y) + b(x,y)\cos\big(\psi(x,y) + \phi(x,y)\big)$$



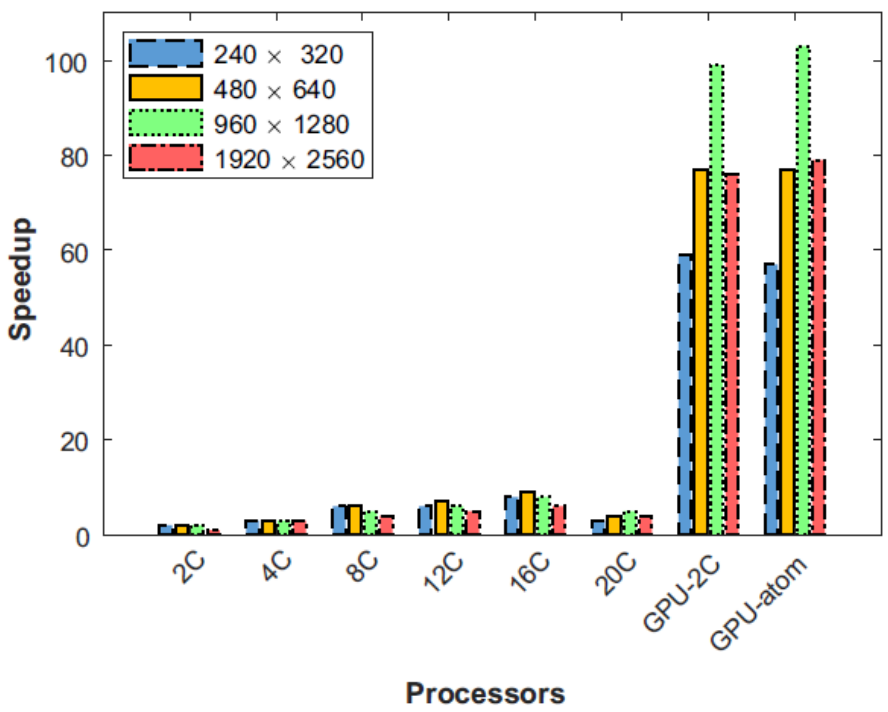


Hernandez-Lopez, F. J., Legarda-Saenz, R., & Brito-Loeza, C. (2021). Parallel algorithm for fringe pattern demodulation. Journal of Real-Time Image Processing, 1-10. https://doi.org/10.1007/s11554-021-01129-4.

# EVALUACIÓN DE VELOCIDAD DEL ALGORITMO PARALELO




Usando precisión de 64-bits (double)



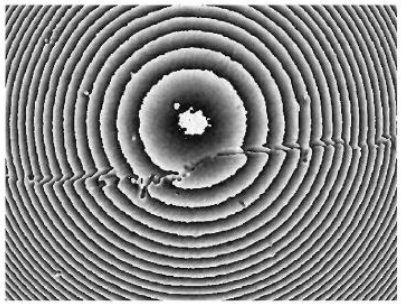

Usando precisión de 32-bits (float)

El algoritmo paralelo alcanza un speedup de **9x** usando multi-core y **103x** usando la GPU, con respecto al algoritmo secuencial.

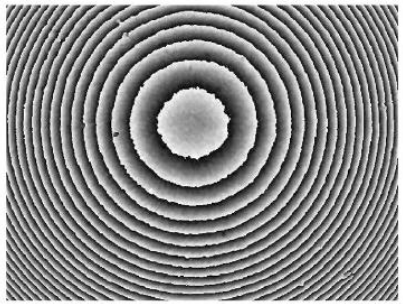
# PARALELIZACIÓN DE UN ALGORITMO PARA EL FILTRADO DE FASES ENVUELTAS

$$(\bar{z}_{\mathrm{re}}, \bar{z}_{\mathrm{im}}) = \arg\min_{\boldsymbol{x}\in\Omega} F(\boldsymbol{x})$$

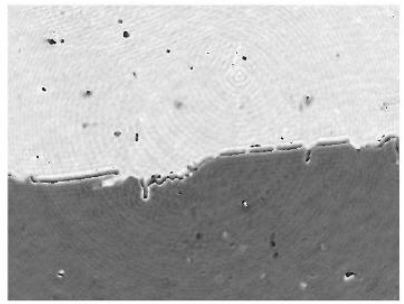
$$F = \frac{\lambda_1}{2}\int_{\Omega} (\bar{z}_{\mathrm{re}} - \hat{z}_{\mathrm{re}})^2 \mathrm{d}\Omega + \frac{\lambda_2}{2}\int_{\Omega} (\bar{z}_{\mathrm{im}} - \hat{z}_{\mathrm{im}})^2 \mathrm{d}\Omega + \frac{\lambda_3}{2}\int_{\Omega} \left((\bar{z}_{\mathrm{re}})^2 + (\bar{z}_{\mathrm{im}})^2 - 1\right)^2 \mathrm{d}\Omega + \int_{\Omega} |\nabla \bar{z}_{\mathrm{re}}| \mathrm{d}\Omega + \int_{\Omega} |\nabla \bar{z}_{\mathrm{im}}| \mathrm{d}\Omega$$



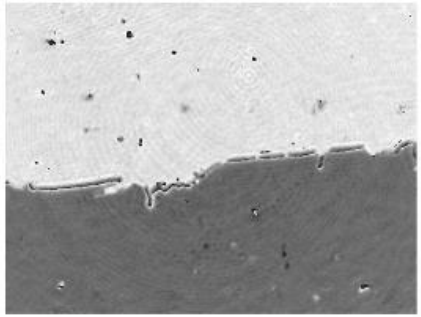
Fase con transport.



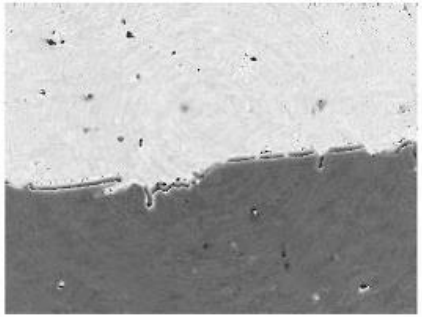
Transportadora



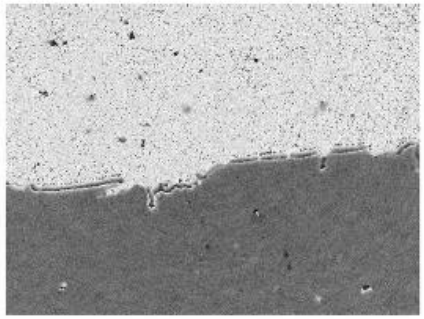
Diferencias



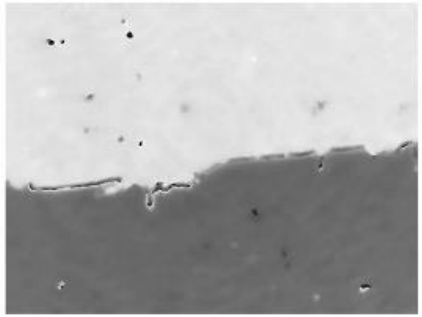
Ströbel



BM3D



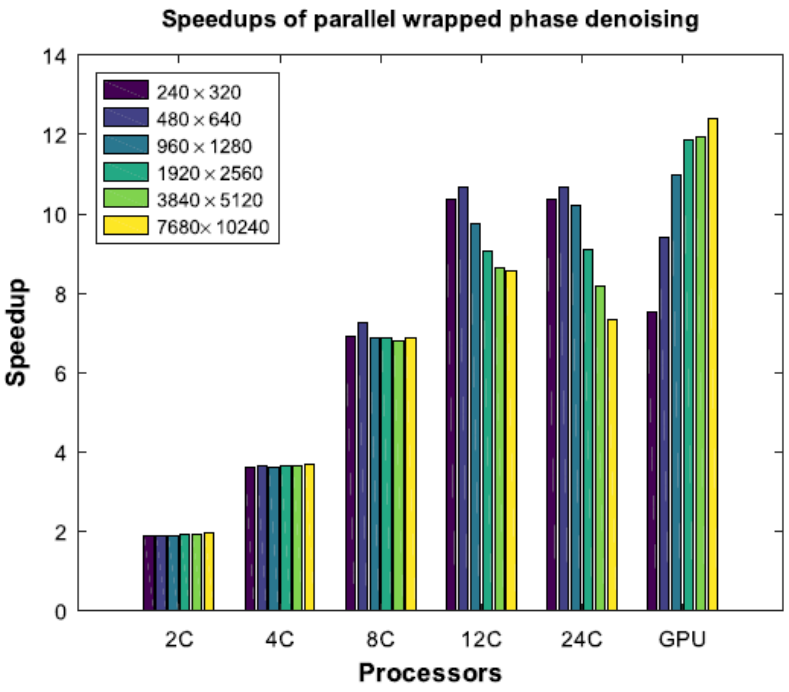
Non-Local Means



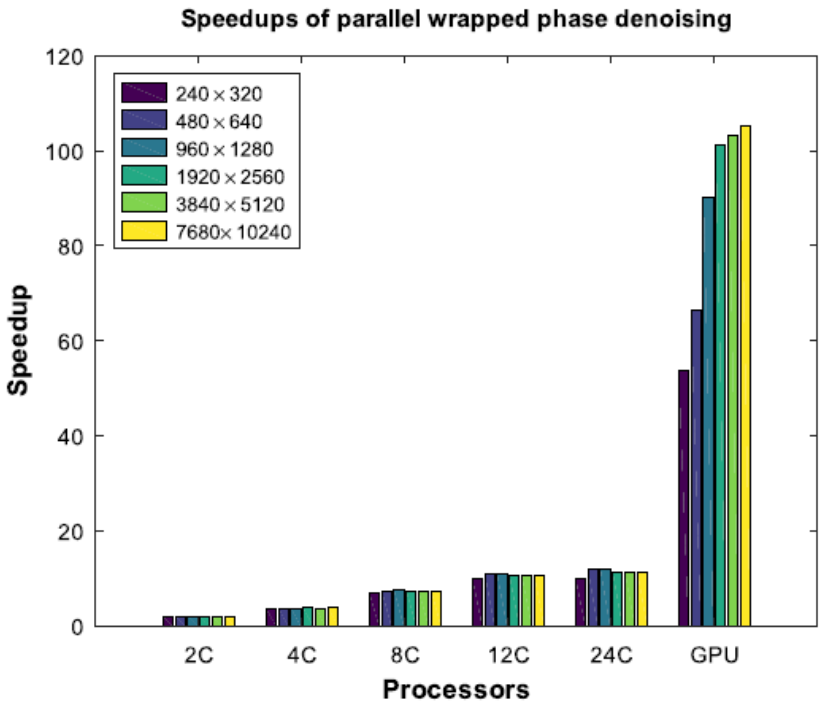
May-Cen et. al

May-Cen, I. D. J., Hernandez-Lopez, F. J., Legarda-Sáenz, R., & Brito-Loeza, C. (2023). Parallel algorithm for wrapped phase denoising. *Journal of Real-Time Image Processing*, *20*(4), 68.

# SPEEDUPS – FP64 Y FP32



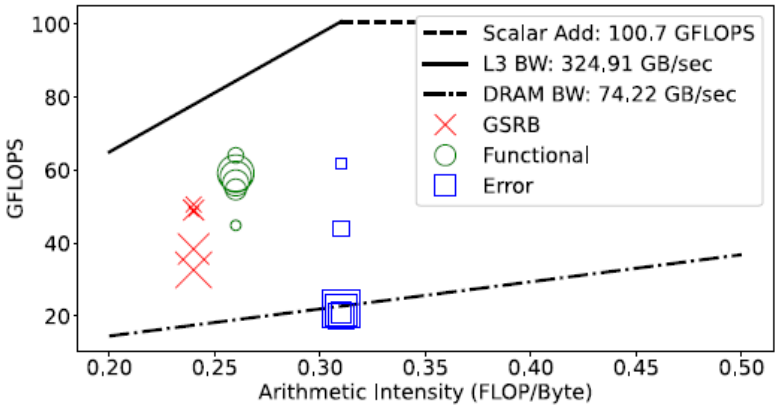

Usando precisión de 64-bits (double)



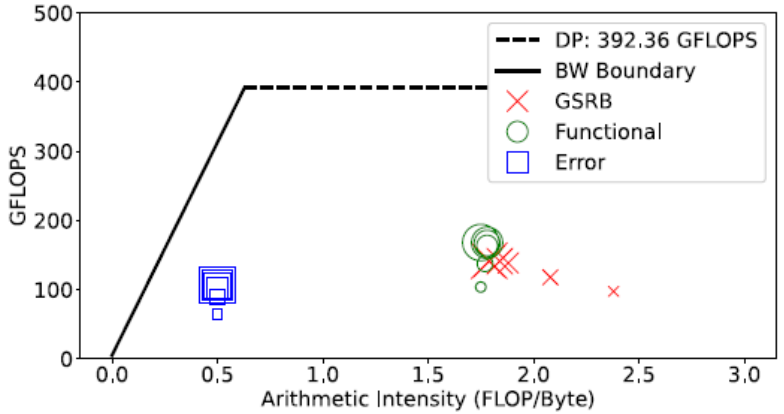

Usando precisión de 32-bits (float)

El algoritmo paralelo alcanza un speedup de **12x** usando multi-core y **110x** usando la GPU con FP32, con respecto al algoritmo secuencial.

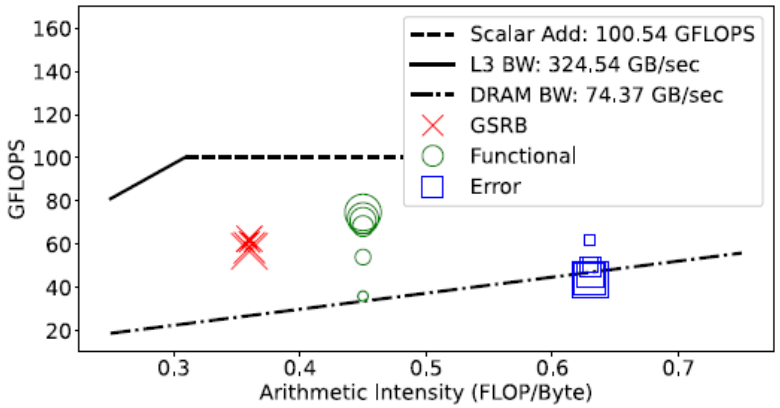
# COMPARANDO GFLOPS




MCore, FP64

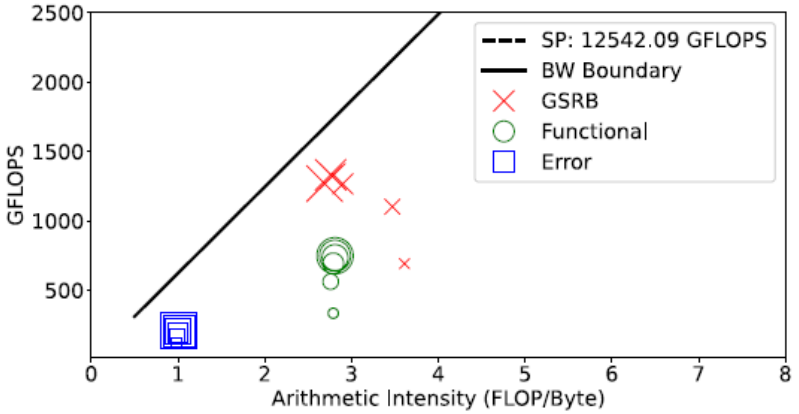



GPU, FP64




MCore, FP32

Intel Advisor



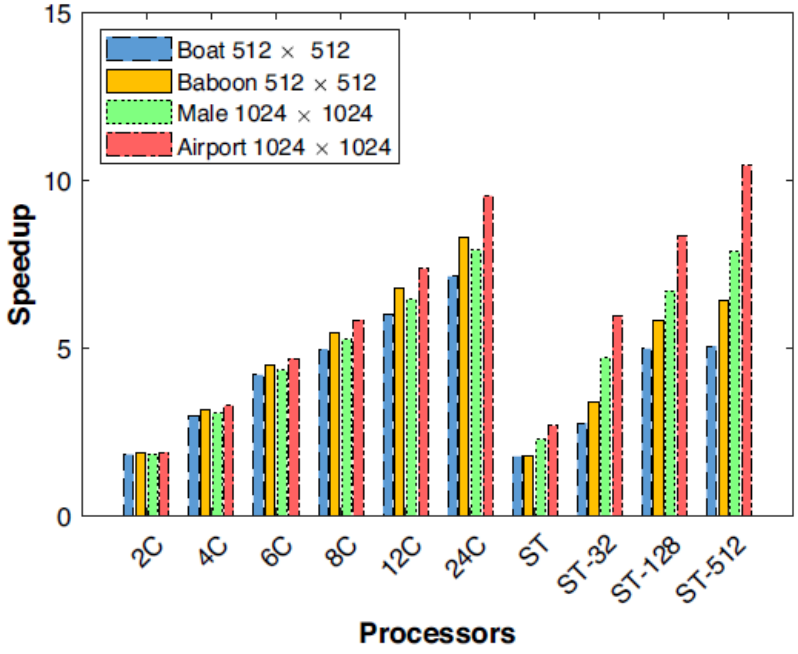

GPU, FP32

Nvidia Nsight Compute

# PARALELIZACIÓN DE UN ALGORITMO DE COMPRESIÓN FRACTAL DE IMÁGENES

Hernandez-Lopez, F. J., Muñiz-Pérez, O. (2022). Parallel fractal image compression using quadtree partition with task and dynamic parallelism. J Real-Time Image Proc. https://doi.org/10.1007/s11554-021-01193-w.

# SPEEDUP MULTI-CORE VS GPU



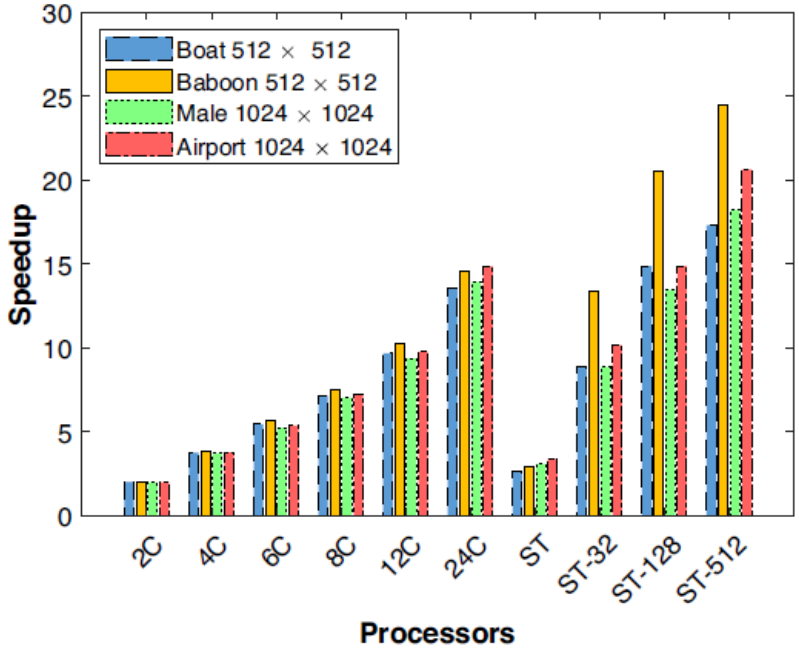
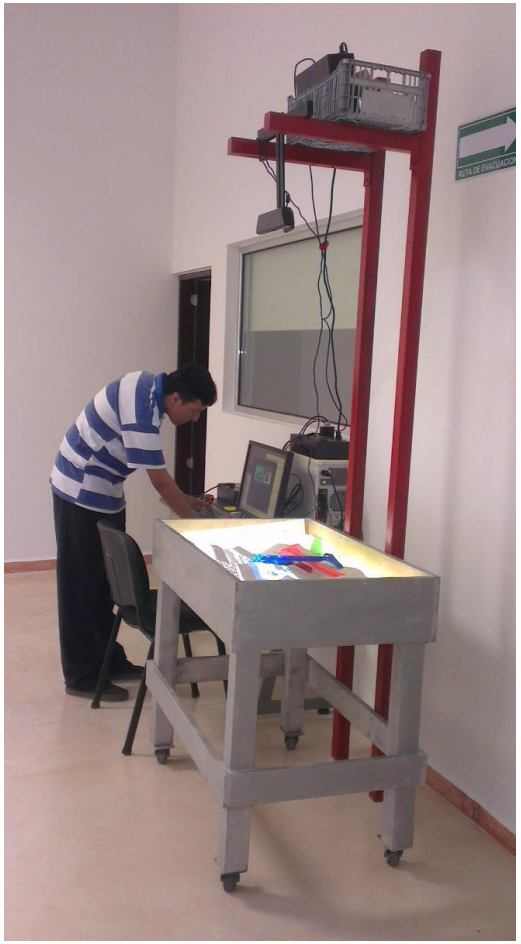




**Fig. 2** Evaluation of speedups of our FICQP implementation, using the multi-core CPU (from 2C to 24C), and the GPU with streams and a different number of TPB (from 32 to 512) in the server: **a** with `tol` = 20; **b** with `tol` = 5

# MESA DE ARENA CON REALIDAD AUMENTADA

**Tesis de Lic. en Ciencias de la Comp. (Nov/2021)**

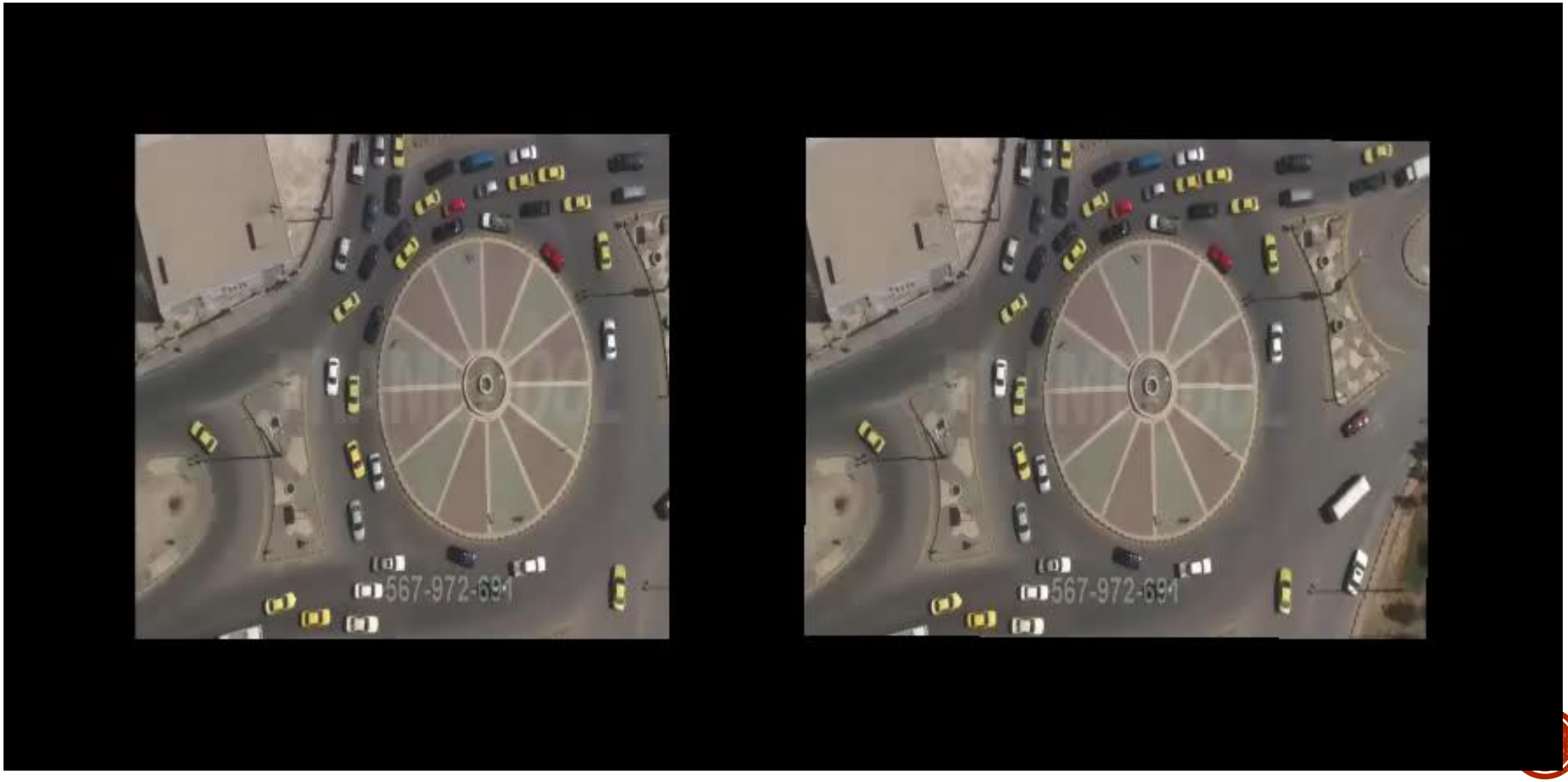
Daniel Israel Ceballos Uc



# ESTIMACIÓN DEL FLUJO VEHICULAR

**Tesis de Maestría en Ing. del ITM (Ene/2019)**

Rafael Puerto Valladares

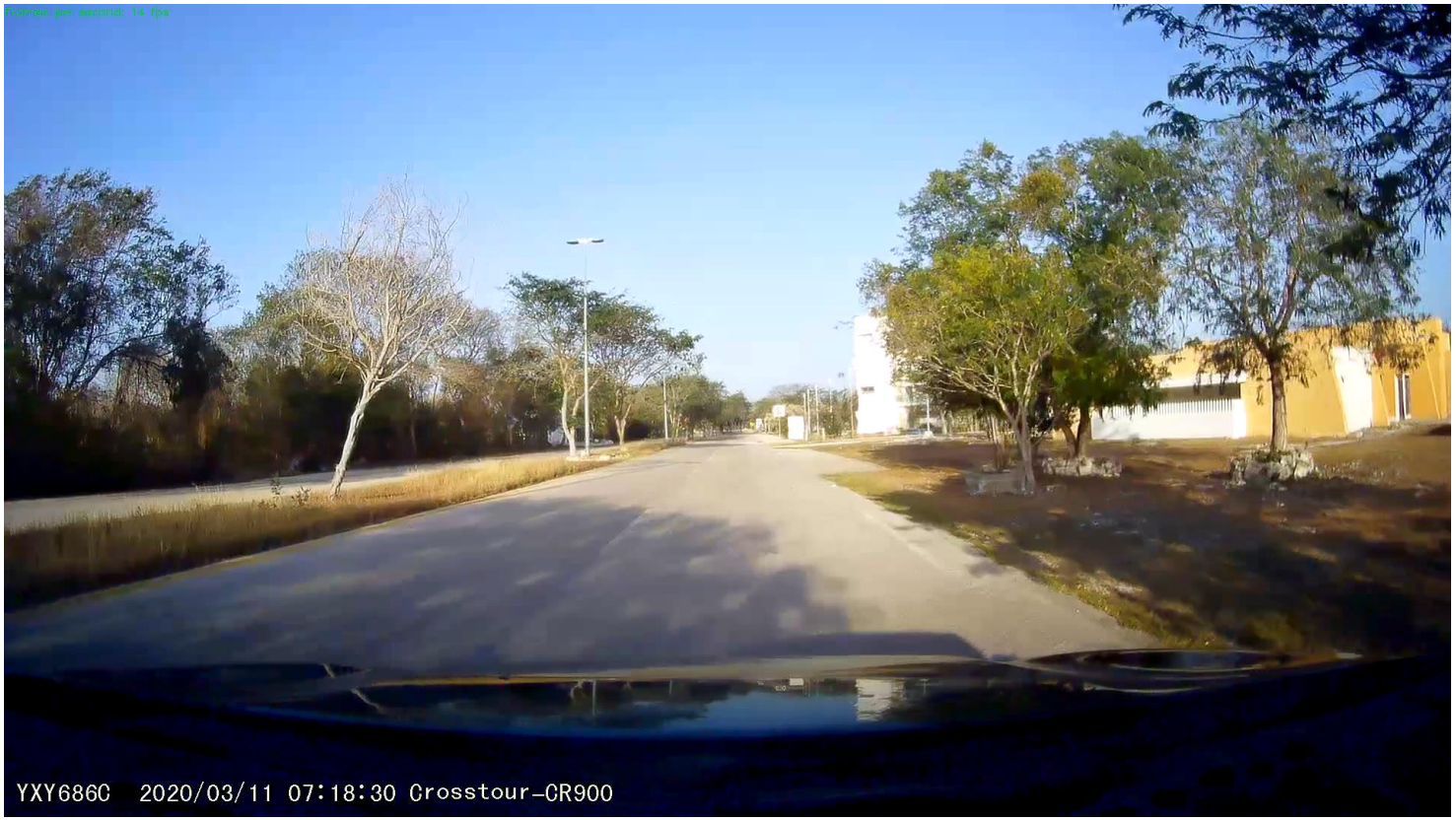


# RECONOCIMIENTO DE SEÑALES DE TRÁNSITO

**Tesis de Maestría en Ing. del ITM**

Edwin Julián González Correa

22/Ago/2019-16/Jun/2022



# DETECCIÓN DE BACHES

**Tesis de Maestría en Cómputo Estadístico de CIMAT-Mty**

Enrique Eduardo Cortés Montes

18/Ene/2021-12/Ago/2022

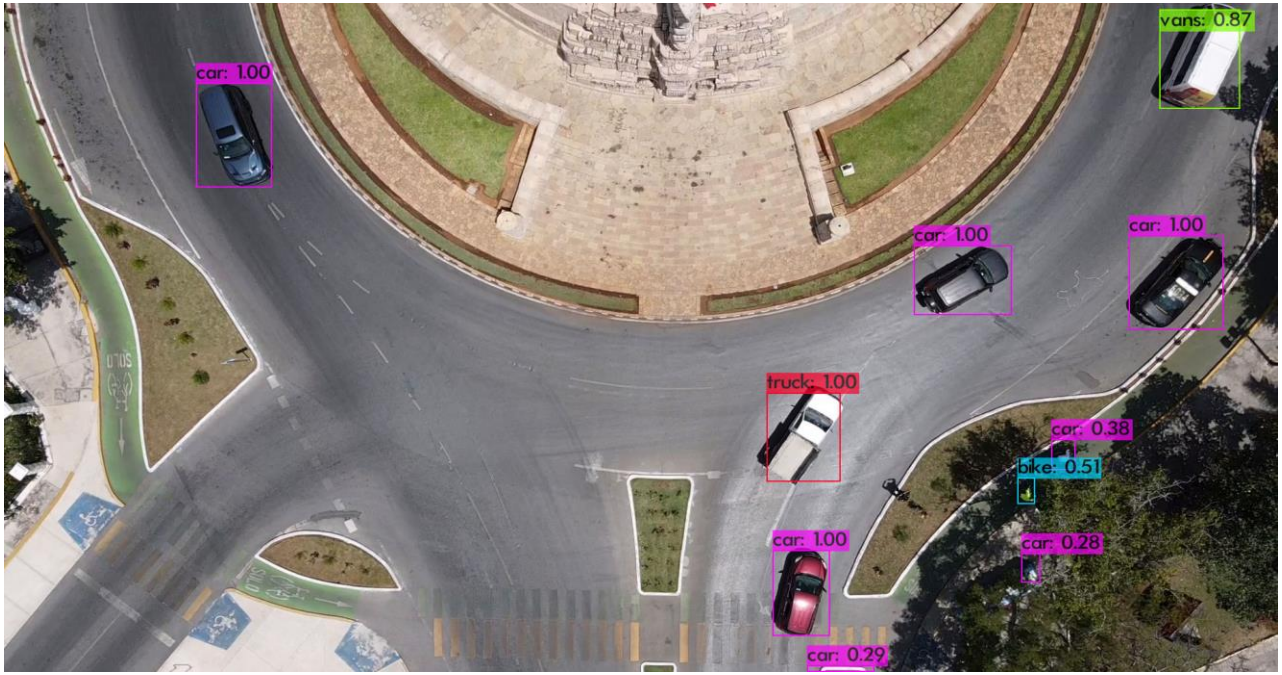
CIMAT

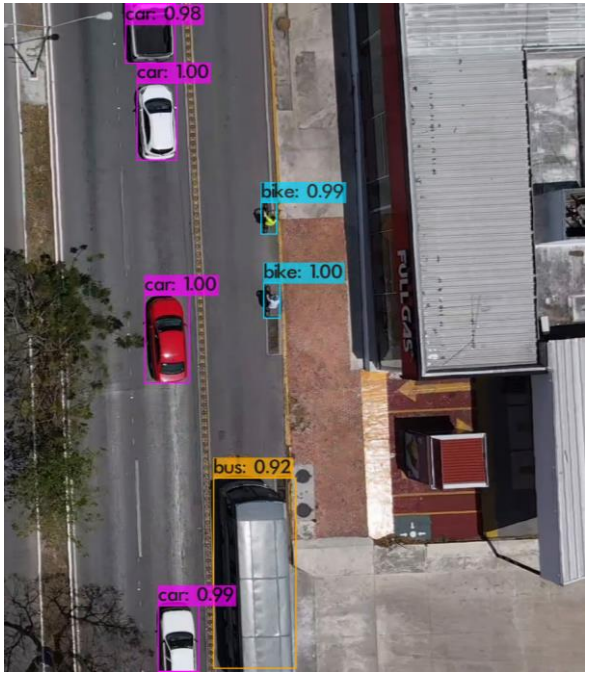
# DETECCIÓN Y CLASIFICACIÓN DE VEHÍCULOS A PARTIR DE IMÁGENES AÉREAS

**Tesis de Ing. en Sistemas Comp. del ITM**

Ángel de Jesús Can Pech

06/Ene/2022 - Actualmente



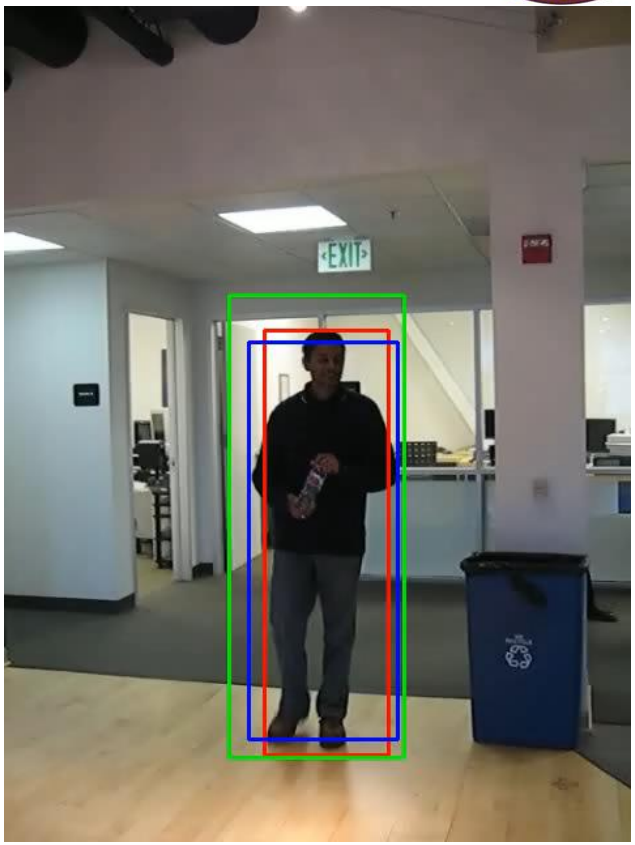



# SEGUIMIENTO DE OBJETOS EN VIDEO

**Tesis de Maestría en Ing. del ITM**

Andrés Ely Pat chan

27/Oct/2021 - Actualmente

# CONTEO AUTOMÁTICO DE VEHÍCULOS EN MOVIMIENTO

**Tesis de Maestría en Cómputo Estadístico de CIMAT-Mty**

José Armando Salcedo Delgado

21/Oct/2022 - Actualmente

CIMAT

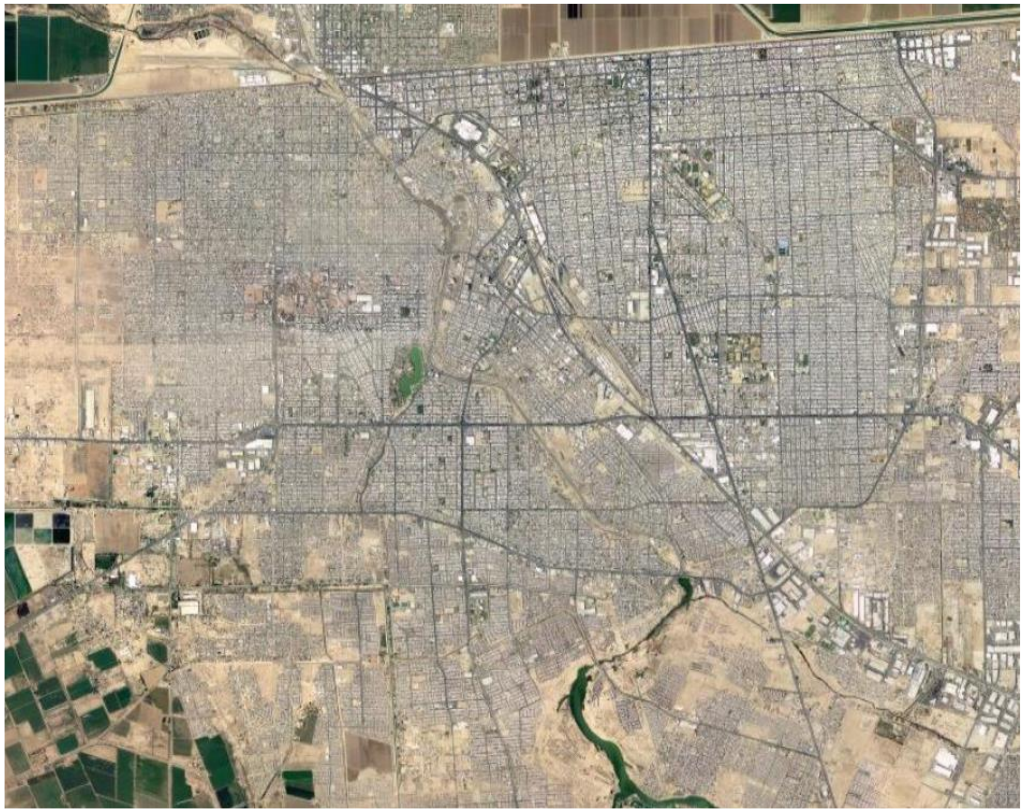
# DETECCIÓN DE ANTIESPACIOS URBANOS

**Tesis de Maestría en Modelación y Optimización de Procesos de CIMAT-Ags**

Jorge Adrián Martínez López

26/Sep/2022-Actualmente

CIMAT



# GRACIAS POR SU ATENCIÓN

Dr. Francisco J. Hernández-López

fcoj23@cimat.mx

WebPage:

www.cimat.mx/~fcoj23