ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

“DISEÑO DE UNA APLICACIÓN INTERACTIVA QUE PERMITA ASISTIR AL CONDUCTOR EN EL RECONOCIMIENTO DE SEÑALES DE TRÁNSITO MEDIANTE PROCESAMIENTO DE IMÁGENES UTILIZANDO SOFTWARE LIBRE Y TECNOLOGÍA BEAGLEBONE”.

Realizado por:
Luis Ernesto Caiza Andrango

Revisado por:
Director: Ing. Eddie Galarza

AGENDA

- Objetivos e Hipótesis
- Descripción de la tarjeta Beaglebone Black
- Descriptores locales y algoritmo SURF
- Desarrollo de la aplicación
- Análisis de resultados
- Conclusiones y recomendaciones

OBJETIVOS

GENERAL

- Diseñar de una aplicación interactiva que permita asistir al conductor en el reconocimiento de señales de tránsito mediante procesamiento de imágenes utilizando software libre y tecnología BEAGLEBONE.

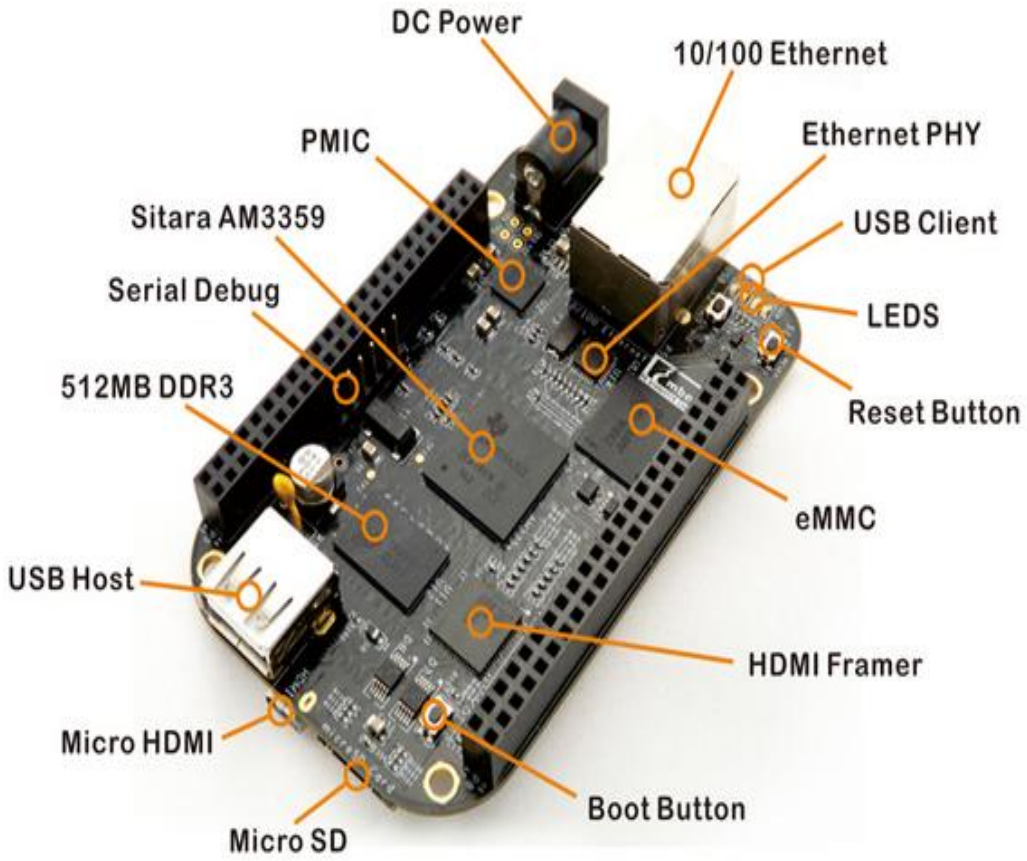
ESPECÍFICOS

- Investigar sobre el procesamiento de imágenes, los diferentes métodos de procesamiento, diferentes algoritmos y aplicaciones.
- Estudiar el algoritmo SURF para el procesamiento de imágenes.
- Estudiar sobre el funcionamiento de la tarjeta Beaglebone Black como unidad de procesamiento de imágenes y cómo tarjeta de adquisición de datos.
- Realizar pruebas con la aplicación funcionando para analizar los resultados

HIPÓTESIS

El diseñar una aplicación interactiva para el reconocimiento de señales de tránsito de velocidad mediante procesamiento de imágenes utilizando software libre y a tecnología Beaglebone Black, será factible implementarla en los vehículos y permitirá que el conductor preste atención en la vía al momento de conducir.

TARJETA BEAGLEBONE BLACK



Diseñada para aplicaciones que trabajen directamente sobre el procesador.

Compatible con algunas distribuciones de Linux como Angstrom, Debian, Ubuntu, también con Android.

GPIO, Entradas analógicas, Puertos PWM, comunicación serial.

OPENCV

- Es una biblioteca para C, C++, Python y Java, siendo compatible con Windows, Linux, Mac OS, iOS y Android.
- Da la posibilidad de realizar procesamiento de imágenes y visión por computador.
- OpenCV posee estructuras básicas para el procesamiento de imágenes.

DESCRIPTORES LOCALES

Un descriptor local es un vector de características que es calculado sobre una pequeña región de interés de la imagen

SIFT: (David Lowe, 1999): transforma una imagen en un conjunto de descriptores.

ASIFT: Mejora a SIFT en cuanto al: el zoom, el ángulo de la cámara, etc.

ORB: Está basado en los siguientes algoritmos; FAST para la detección de Puntos de interés. Y BRIEF para la extracción de descriptores.

SURF: Speeded-Up Robust Features, (Herbert Bay, 2006):

ALGORITMO SURF

- *Detección de puntos de interés.-* Se basa en la matriz Hessiana por el buen rendimiento en cuanto a la velocidad de cálculo y la precisión.

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix}$$

- Donde $L_{xx}(\rho,\sigma)$ es la convolución de la derivada parcial de segundo orden de la Gaussiana $\frac{\theta^2}{\theta x^2} g(\sigma)$ con la imagen $I(x,y)$ en el punto p

ALGORITMO SURF

- Para aumentar la velocidad en el cálculo, el descriptor utiliza filtros de caja. Estos filtros aproximan los valores utilizando imágenes integrales.

$$I_{\Sigma}(p) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i,j)$$

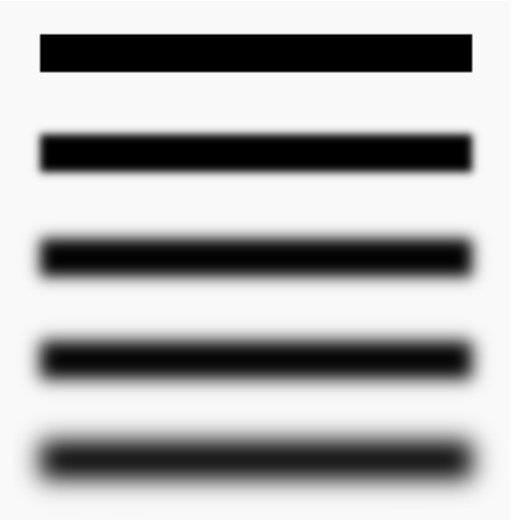


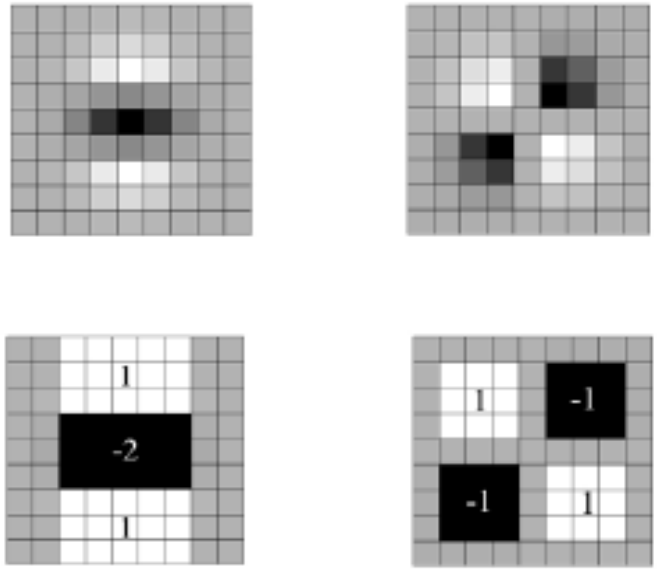
imagen integral $I_{\Sigma}(p)$

- Presentan una serie de limitaciones como la necesidad de discretización, efecto aliasing, etc.

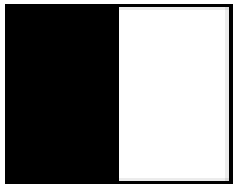
ALGORITMO SURF

Las aproximaciones de las *derivadas parciales* $Dxx, Dxy\ y\ Dyy$

$$Det(Haprox) = D_{xx}D_{yy} - \left(\omega D_{xy}\right)^2$$



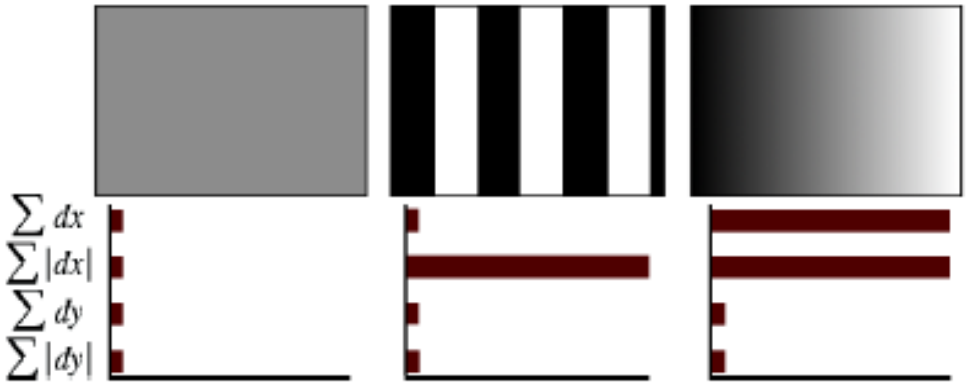
ALGORITMO SURF



- *Asignación de orientación.-* Calcular la respuesta de la dirección *x* e *y*

- *Extracción del descriptor.-* En este paso lo primero es realizar la construcción de una región cuadrada alrededor del punto de interés. Las respuestas dx y dy son ponderadas con una Gaussiana en el punto de interés.

$$v = (\Sigma dx. \Sigma dy. \Sigma|dx|, \Sigma|dy|)$$



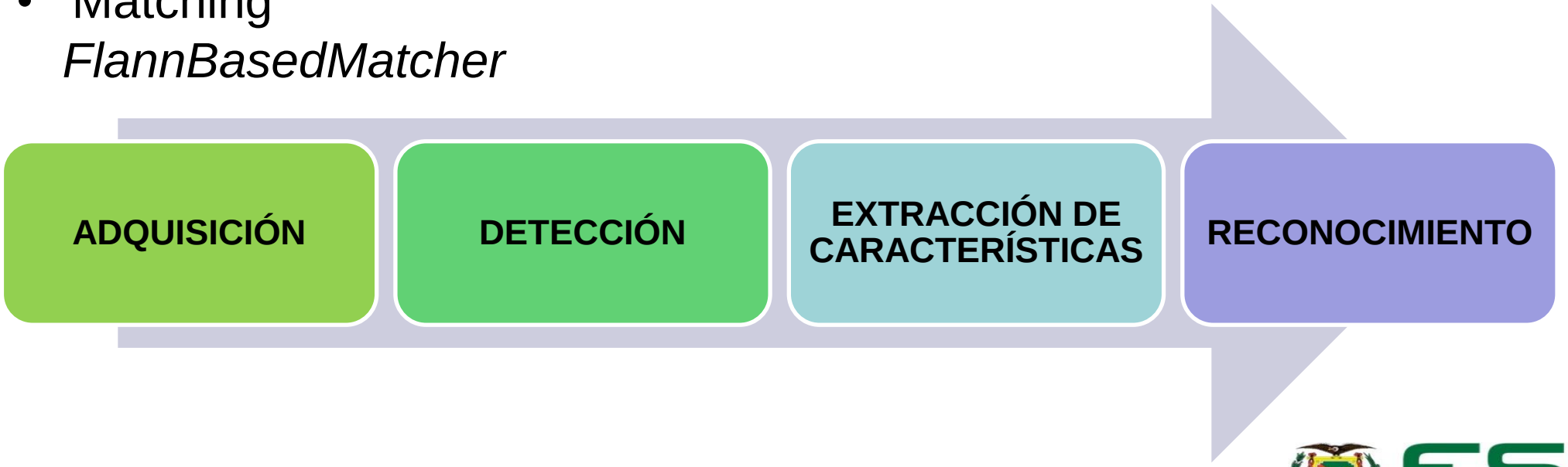
- *Matching.*

DESARROLLO DE LA APLICACIÓN

- Adquisición de imágenes en tiempo real clase Mat y clase IplImage.

- Detección de puntos característicos
  SurfFeatureDetector detector(minhessian);
  SurfDescriptorExtractor extractor;

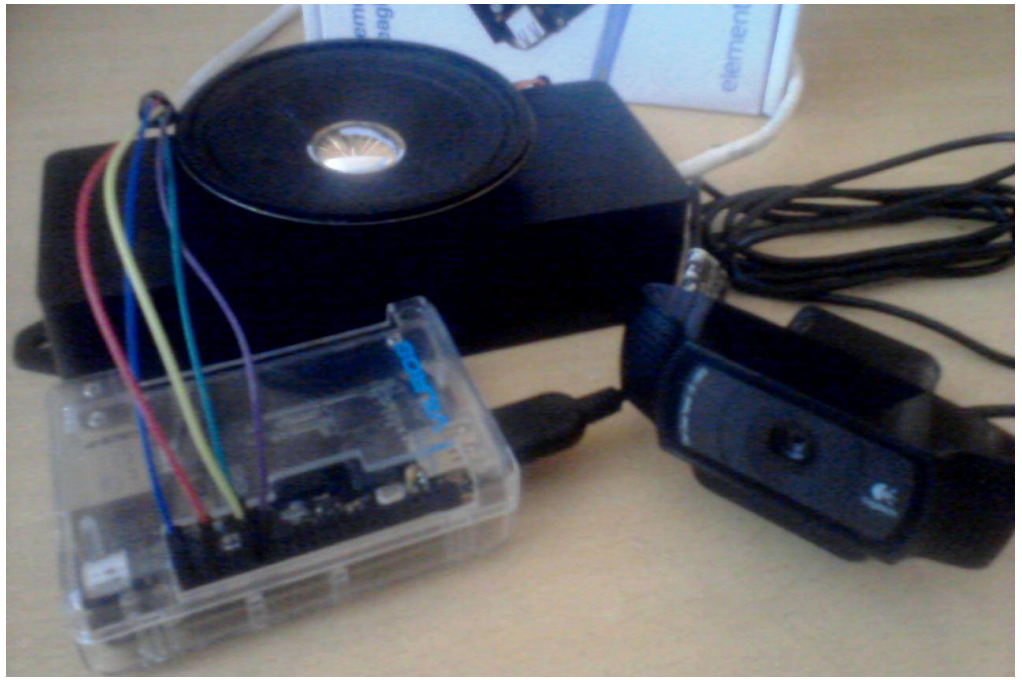
- Matching
  FlannBasedMatcher



DESARROLLO DE LA APLICACIÓN

Control de los Puertos GPIO de la tarjeta Beaglebone Black

- Exportar el pin: echo “pin” > */sys/class/gpio/export*
- Definir el pin como entrada o salida: *echo “in/out” > /sys/class/gpio/gpiopin/ direction*
- Fijar valor lógico: *echo “1/0” > /sys/class/gpio/gpiopin/value*
- Para controlar los pines desde C++ se deben fijar las bibliotecas de cabecera *GPIO/GPIOManager.h y GPIO/GPIOConst.h.*



ANÁLISIS DE RESULTADOS

Detección de Puntos característicos

Señal de 50km/h

Valor del Hessiano	Resolución de la imagen en pixeles			
	250X250	370X370	480X480	640X640
1000	903	2485	3160	3180
2000	741	1378	1378	1728
3000	730	981	1225	1225
4000	403	735	1176	1225
5000	266	541	820	990

Señal de 80km/h

Valor del Hessiano	Resolución de la imagen en pixeles			
	250X250	370X370	480X480	640X640
1000	903	3403	4095	5253
2000	666	1378	1770	1740
3000	624	946	1485	1378
4000	601	746	931	978
5000	295	503	676	796

Señal de 100km/h

Valor del Hessiano	Resolución de la imagen en pixeles			
	250X250	370X370	480X480	640X640
1000	780	946	965	1716
2000	666	561	580	1485
3000	564	528	542	739
4000	435	478	528	560
5000	270	308	388	492

ANÁLISIS DE RESULTADOS

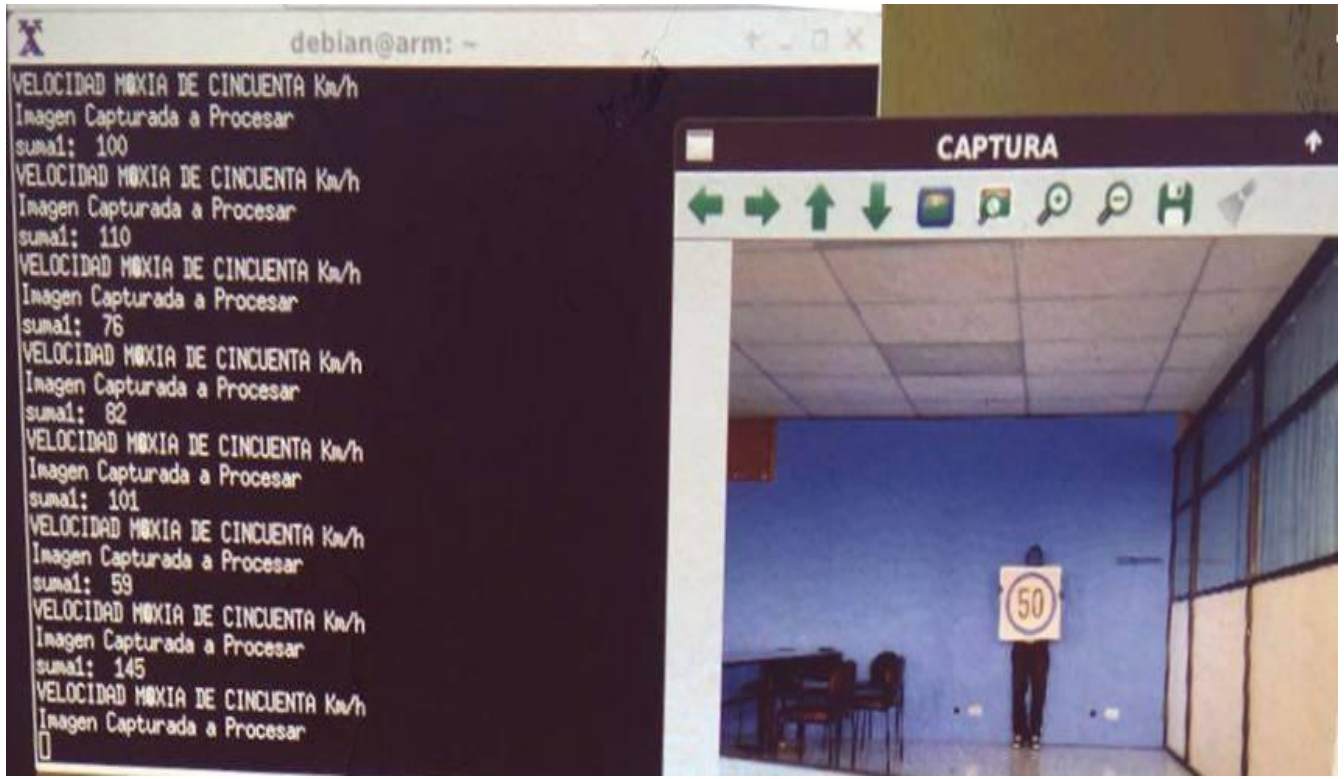
Tiempos de Ejecución del algoritmo SURF

Computador

Valor Hessiano	Tiempo(s) (640x480)	Tiempo(s) (1024x748)
1000	0,130652	0,134073
2000	0,131127	0,134626
3000	0,126734	0,123474
4000	0,120586	0,113969
5000	0,118219	0,114686

Beaglebone Black

Valor Hessiano	Tiempo(s) (640x480)	Tiempo(S) (1024x748)
1000	4,450736	4,550736
2000	4,806269	4,642569
3000	3,534743	3,493743
4000	2,739696	2,823696
5000	2,306864	2.463864

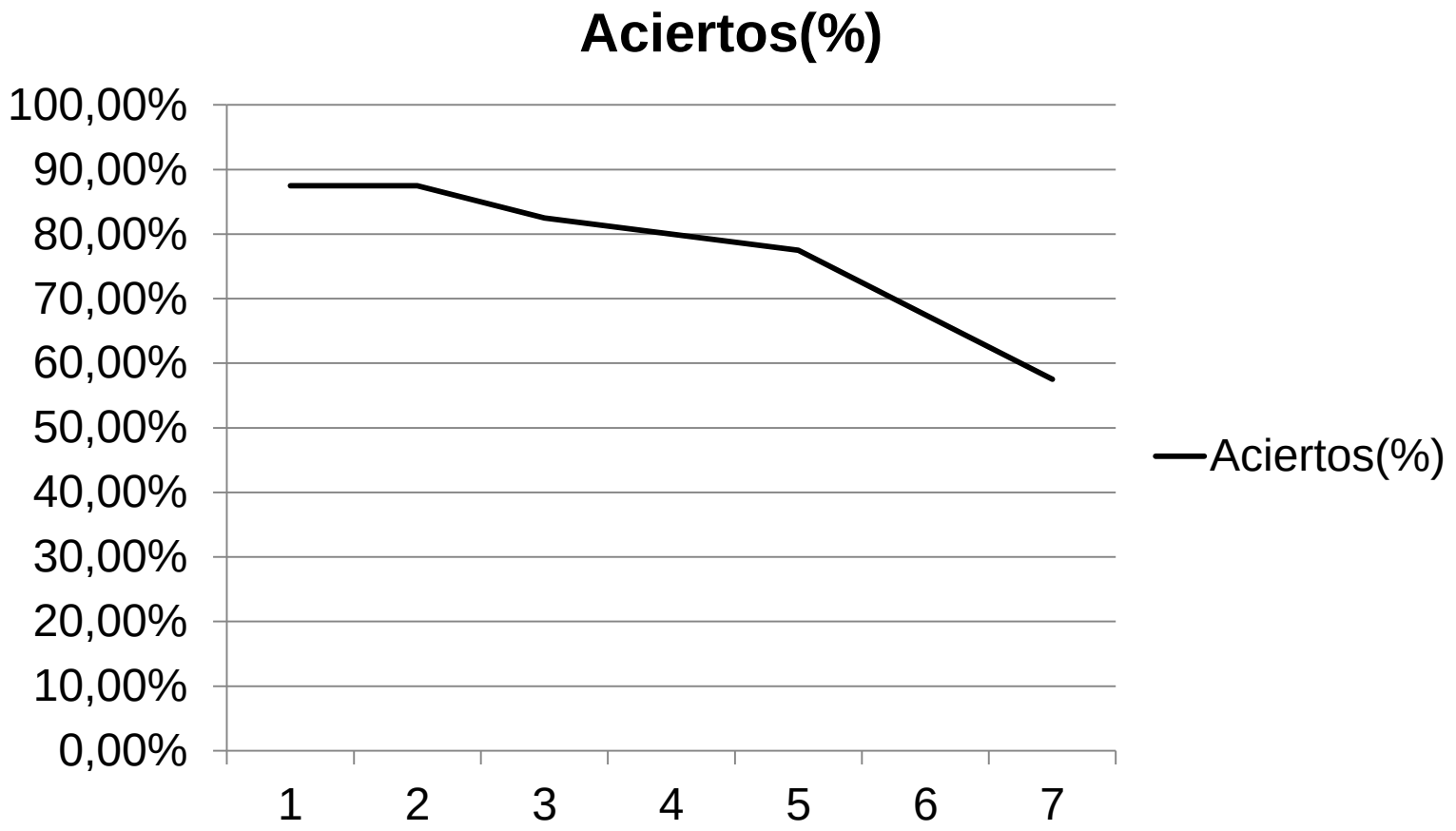


ANÁLISIS DE RESULTADOS

Reconocimiento de objetos con iluminación baja

Resolución 640x480

Distancia(m)	Aciertos(%)
1	87,5%
2	87,5%
3	82,5%
4	80%
5	77,5%
6	67,5%
7	57,5%

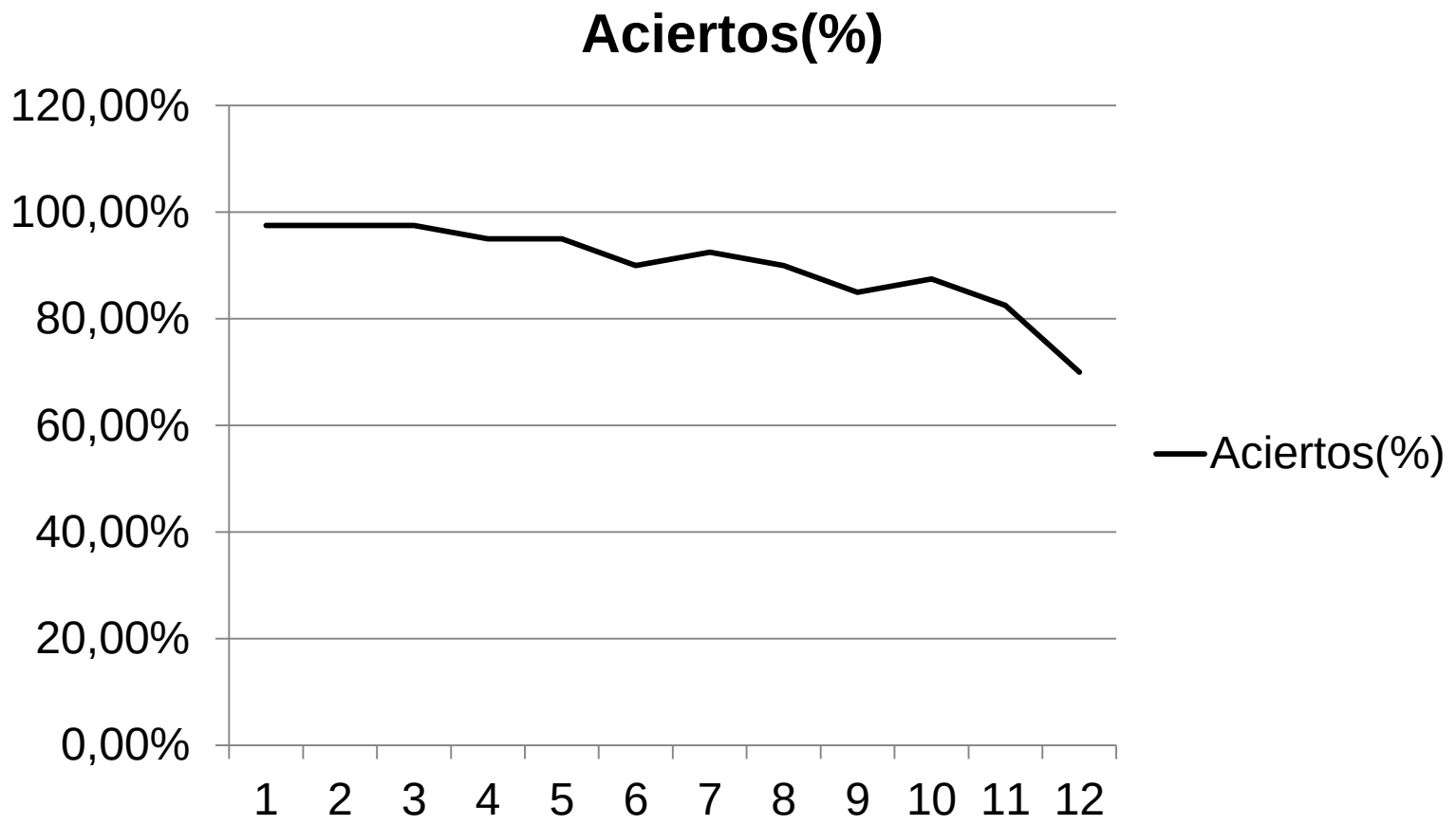


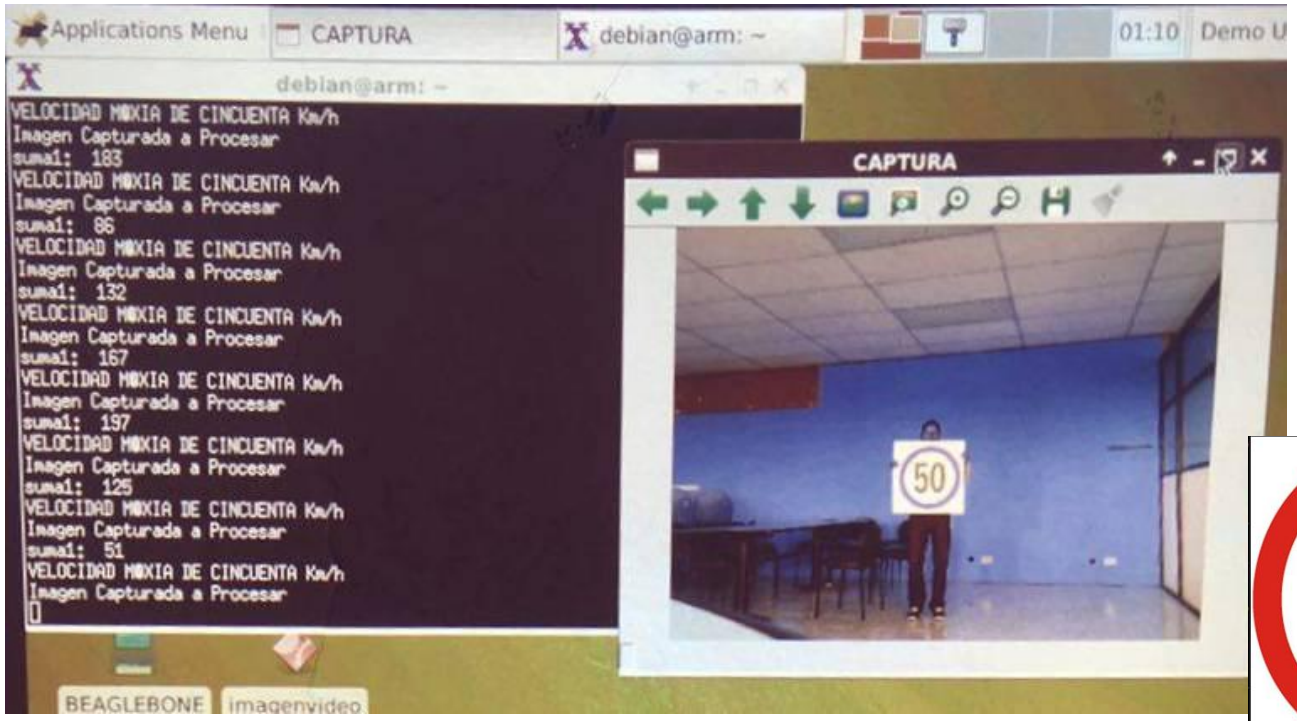
ANÁLISIS DE RESULTADOS

Reconocimiento de objetos con iluminación baja

Resolución 1024x748

Distancia(m)	Aciertos(%)
1	97,5%
2	97,5%
3	97,5%
4	95%
5	95%
6	90%
7	92,5%
8	90%
9	85%
10	87,5%
11	82,5%
12	70%




Applications Menu
CAPTURA
debian@arm: ~
01:10
Demo U
debian@arm: ~
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 183
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 86
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 132
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 167
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 197
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 125
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
sumai: 51
VELOCIDAD MAXIA DE CINCUENTA Km/h
Imagen Capturada a Procesar
CAPTURA
50
BEAGLEBONE
imagenvideo

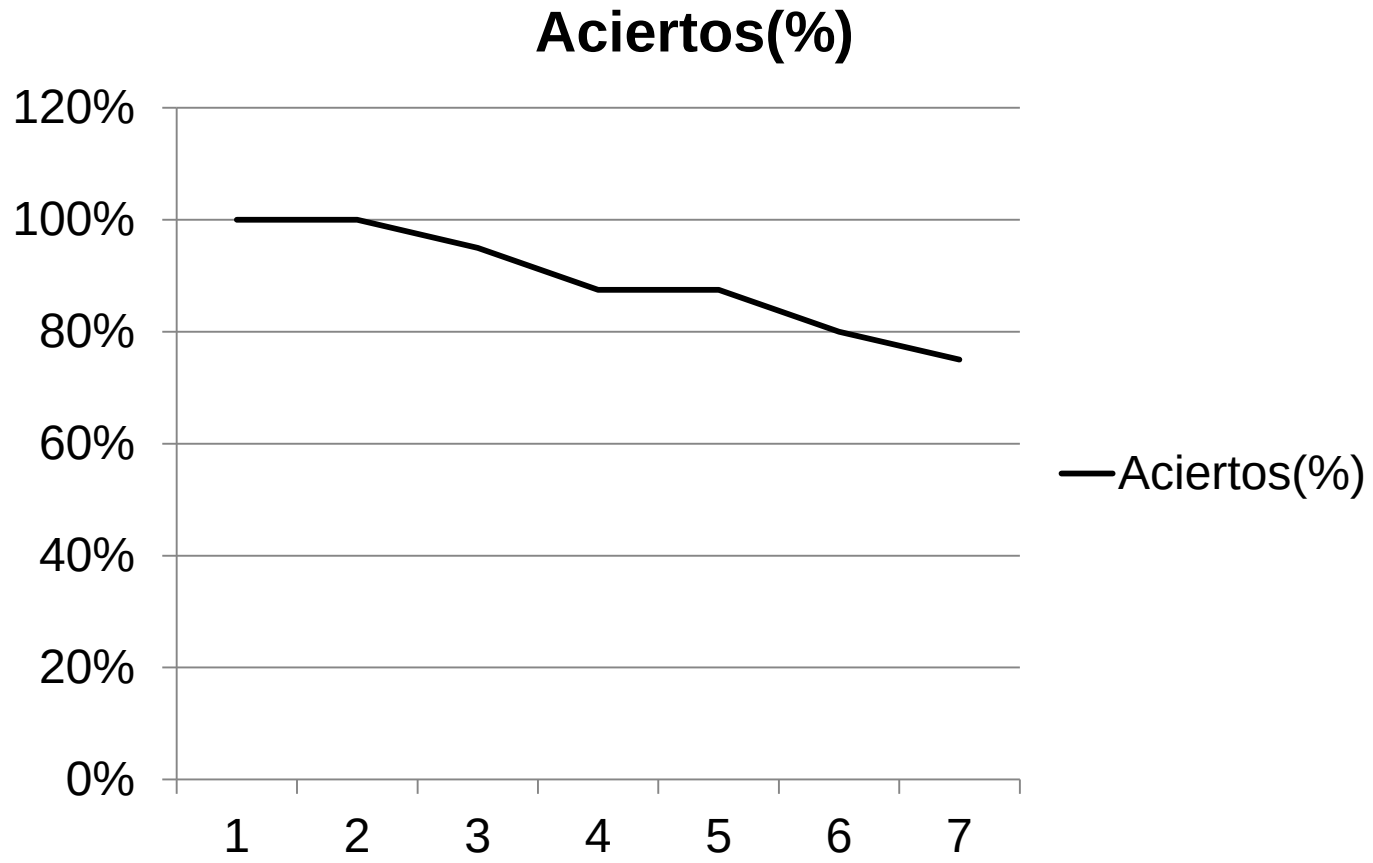

50
PTA-626

ANÁLISIS DE RESULTADOS

Reconocimiento de objetos con iluminación alta

Resolución 640x480

Distancia(m)	Aciertos(%)
1	100%
2	100%
3	95%
4	87,5%
5	87,5%
6	80%
7	75%

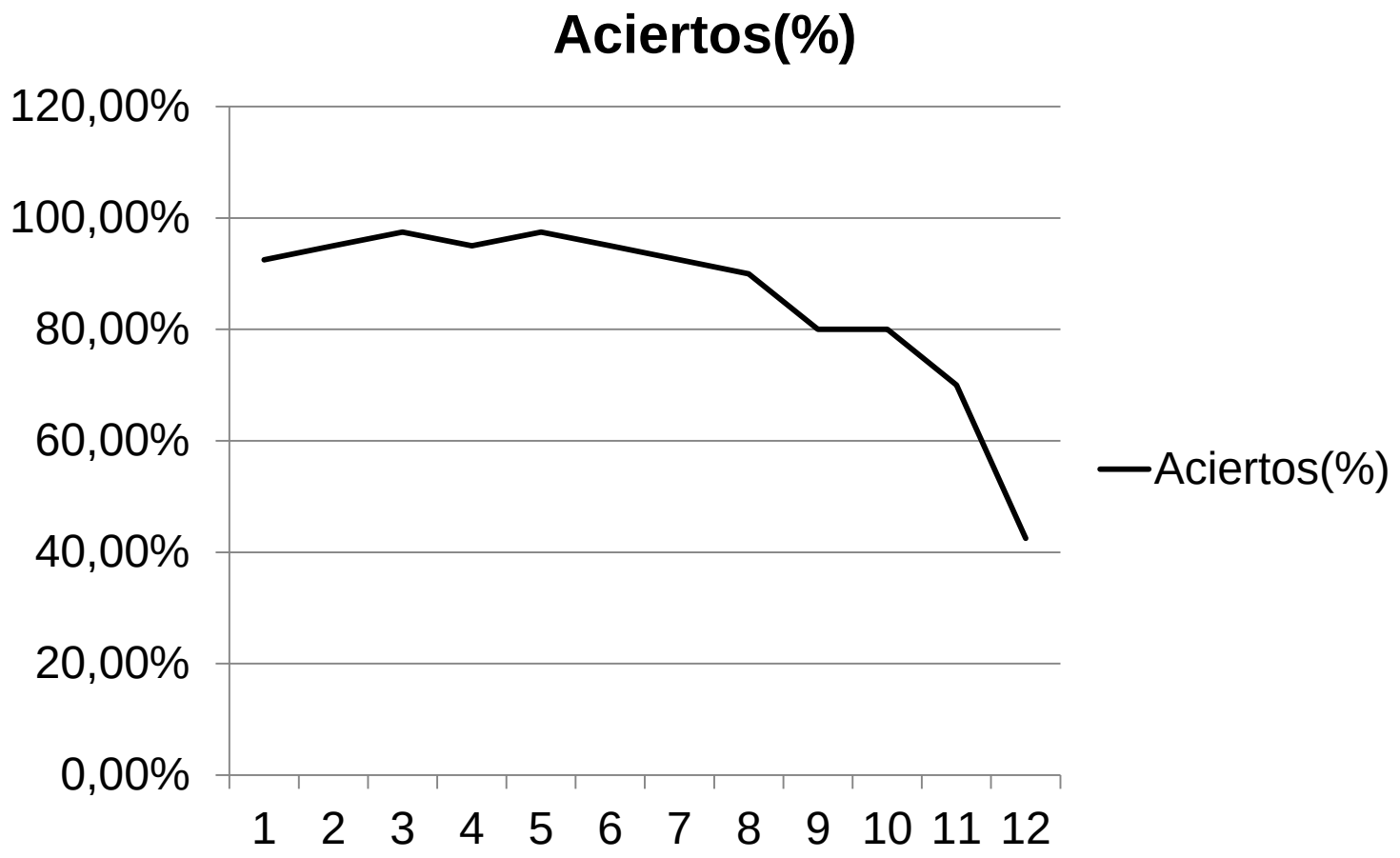


ANÁLISIS DE RESULTADOS

Reconocimiento de objetos con iluminación alta

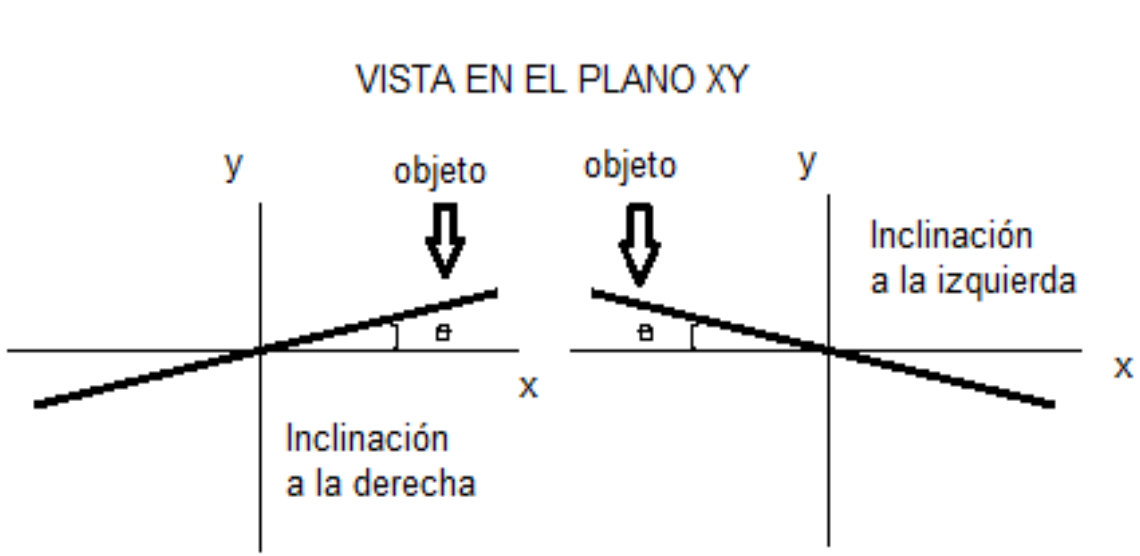
Resolución 1024x748

Distancia(m)	Aciertos(%)
1	92,5%
2	95%
3	97,5%
4	95%
5	97,5%
6	95%
7	92,5%
8	90%
9	80%
10	80%
11	70%
12	42,5%



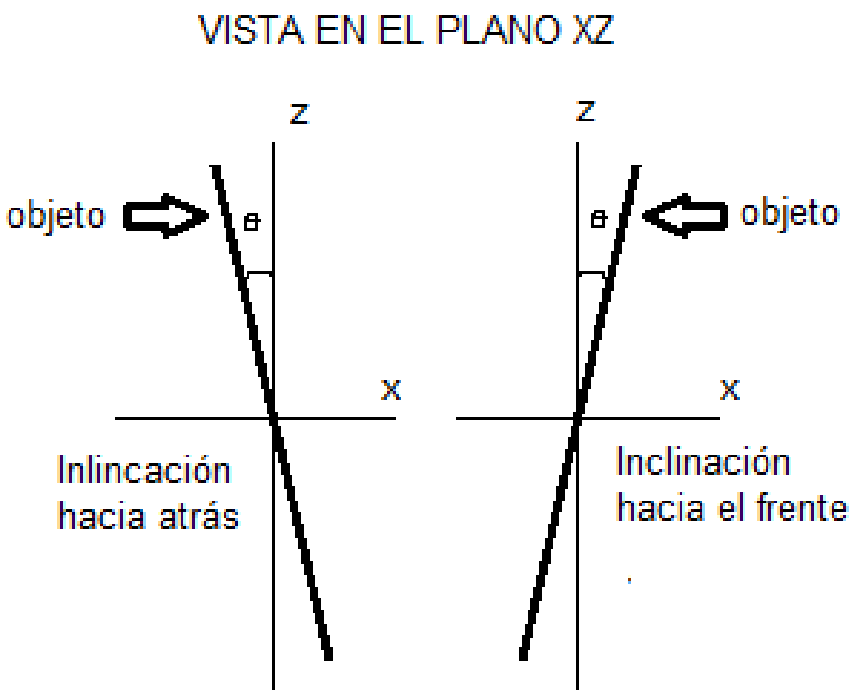
ANÁLISIS DE RESULTADOS

Inclinación lateral



ANÁLISIS DE RESULTADOS

Inclinación frontal



CUMPLIMIENTO DE LA HIPÓTESIS PLANTEADA

- La tarjeta beaglebone Black al ser un microcomputador permite trabajar en el área de procesamiento de imágenes.
- Al no tener una cámara especifica para trabajar con la Beaglebone Black se debe buscar la más adecuada dependiendo de la aplicación que se realizará.
- La detección de señales de tránsito por medio del algoritmo SURF se realiza de una manera óptima pero con un retardo apreciable.
- Existen errores al momento de la identificación y es debido a que los objetos tiene similitudes.
- Se puede controlar los puertos de la tarjeta Beaglebone Black al mismo tiempo que se ejecuta el algoritmo de reconocimiento de objetos desde C++, con la finalidad de activar el módulo de audio y dar una señal sonora cuando se ha reconocido el objeto.

CONCLUSIONES

- El sistema operativo que se instale en la tarjeta Beaglebone Black influye mucho en el rendimiento de software y hardware, en los tiempos de ejecución y la robustez que puede ofrecer.

- La implementación del algoritmo SURF para detección de objetos en tiempo real en tarjeta Beaglebone Black sirve de manera óptima siempre y cuando el proceso para la detección de objetos sea lento, debido a que el algoritmo necesita de muchos recursos de memoria al momento de procesar imágenes y consecuente a esto se demora en el procesamiento.

- En el procesamiento de imágenes mediante el algoritmo SURF se debe tener presente los factores de iluminación, distancia, inclinación y resolución de la imagen capturada, ya que estos afectan de manera directa.

RECOMENDACIONES

- Trabajar con un valor de puntos característicos bajo de unos cincuenta puntos como se hizo en el proyecto, de este modo no tener valores repetidos y asegurando que el reconocimiento sea mejor

- Utilizar una micro-SD de alta velocidad de transferencia de datos, para de este modo mejorar el rendimiento de las aplicaciones que se desean implementar en la tarjeta Beaglebone Black.

- Trabajar con una tarjeta de mayores prestaciones en hardware y software para realizar el procesamiento de imágenes y el reconocimiento de señales de tránsito de tiempo real utilizando el algoritmo SURF.

GRACIAS

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA