Diseño, Análisis e Implementación de un Sistema de Configuración Automático de Parámetros de una Cámara en Soft Real Time

Isabel Andrade [1] Geovanny Soria [2] Daniel Ochoa [3], Ph.D.
Facultad de Ingeniería en Electricidad y Computación [1]
Escuela Superior Politécnica del Litoral (ESPOL)
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil-Ecuador
Isa.bandr@gmail.com [1] geovannyso@gmail.com [2] dochoa@fiec.espol.edu.ec [3]

Resumen

En este trabajo se presenta el análisis, diseño e implementación de un sistema que permite la configuración automática de los parámetros de una cámara en soft real time. El sistema esta implementado sobre V4L2 que es un API de captura de video para Linux que permite la comunicación directa con el kernel del sistema operativo y aprovecha el hardware de la cámara al utilizar hilos que ejecuten sus principales tareas. Cada hilo de ejecución es asignado un procesador, esto disminuye tiempos de ejecución y procesamiento. El brillo, contraste y balance de blancos son manipulados para configurar los parámetros de captura de imágenes. El algoritmo utilizado para esta configuración de parámetros se basa en la comparación del valor promedio de la cada imagen (la suma de los valores de intensidad de cada pixel divido por el total de pixeles) obtenida durante la ejecución y el valor promedio de una imagen de referencia obtenido por inspección visual.

Palabras Claves: software en tiempo real, V4L2, calibración de cámaras.

Abstract

This paper shows the analysis, design and implementation for a system that allows automatic configuration of parameters for a chamber with real time software. This system is based on an API V4L2 video capture for Linux that allows direct communication with the operating system kernel and take advantage of the chamber hardware using threads to execute it's main jobs. Each execution thread is assigned to one processor, therefore decreasing execution and process time. Brightness, contrast and white balance parameters are manually configured for image capturing. The algorithm used for this parameter configuration compares the average value of the image (the sum of the intensity values for each pixel divided by the total pixel) obtained during the execution and a reference image average.

Keywords: soft real time, V4L2, calibration camera.

1. Introducción

Actualmente con la evolución de la tecnología, el procesamiento de imágenes se ha convertido en una disciplina amplia y abarca muchos campos donde se resuelven diversos problemas. Algunos ejemplos son las imágenes obtenidas con fines de diagnósticos médicos, imágenes satelitales obtenidas para realizar exámenes de terreno, etc.

En la práctica, antes de poder extraer alguna información relevante de una imagen es necesario algún tipo de pre-procesamiento, pues muchos algoritmos de procesamiento son susceptibles a la calidad de la imagen. Es decir, que obtendremos mejores resultados de segmentación, detección de bordes, etc. si se tiene una imagen apropiada (imagen con niveles de brillo y contraste que realzan los objetos con los que luego trabajaremos) para procesar.

Algunos de los algoritmos más utilizados en el pre-procesamiento de las imágenes son: eliminación de ruido, suavizado de imagen, realce de bordes, etc. Si bien es cierto que, los algoritmos de pre-procesamiento permiten optimizar una imagen, desarrollarlos involucra un gran esfuerzo y una cantidad de tiempo considerable. Por esto, se recurre a implementaciones existentes de varias librerías de visión por computador que no son necesariamente diseñadas para procesamiento en tiempo real.

Estos algoritmos de pre-procesamiento no siempre permiten obtener los resultados esperados pues algunos no eliminan el ruido por completo y pueden implicar un aumento significativo en los tiempos de ejecución.

El pre-procesamiento podría ser evitado o reducido si un sistema es capaz de capturar imágenes con mejor calidad. Una manera de hacer esto es configurar correctamente los parámetros de la cámara.

En este proyecto se abordan el análisis, diseño e implementación de un sistema que permite realizar la automatización del proceso de configuración de los parámetros de una cámara en soft real time.

En este programa, se obtiene una imagen de referencia que representan imágenes de "buena calidad", luego se ajusta de forma automática los parámetros de la cámara para que las imágenes capturas tengan características similares a la imagen de referencia. El procedimiento se puede realizar de forma periódica para ajustar los parámetros a condiciones de luz cambiante.

2. Materiales y Métodos

Los materiales para la implementación del sistema de configuración automática de parámetros de una cámara se detallan a continuación:

- API de captura de video V4L2.
- Sistema operativo Linux.
- Un computador encargado de alojar las herramientas de software para el funcionamiento del sistema.
- Cámara web.

La metodología utilizada en la implementación del sistema se basa en la utilización de un API de bajo nivel V4L2, para la captura de imágenes. Este permite el acceso a los dispositivos de captura y una configuración dinámica de sus parámetros.

Esta implementación se basa además en el uso de técnicas de programación concurrente y afinidad del CPU, con el objetivo distribuir equitativamente los procesos entre los procesadores mejorando el rendimiento del procesador y reduciendo el tiempo de procesamiento. [1]

3. Diseño Propuesto

La calibración de los parámetros de la cámara está basada en el trabajo de Harsh Nanda propuesto en el artículo: "Practical calibrations for a real-time digital onmidirectional camera" [2]en el cual se presenta un método simple y eficaz para ajustar automáticamente los parámetros de brillo, contraste y balance de blancos en una cámara y luego propagar esos valores en el resto de cámaras y así suavizar los límites de una imagen panorámica obtenida por la unión de las capturas de todas las cámaras.

A este algoritmo se le dio un enfoque diferente para que pueda ser aplicado a la implementación de nuestro proyecto, es por eso que este punto se realiza un análisis del diseño del sistema el cual consiste en obtener una imagen de referencia de "buena calidad", es decir con brillo y contraste adecuado, obtenidas por inspección visual, luego se comparan los valores promedios de la imagen de referencia con la imagen capturada por el sistema y de ser necesarios se ajustan de forma automática los parámetros de la cámara hasta que las imágenes capturadas tengan características similares a la imagen de referencia como se muestra en la figura 1 .

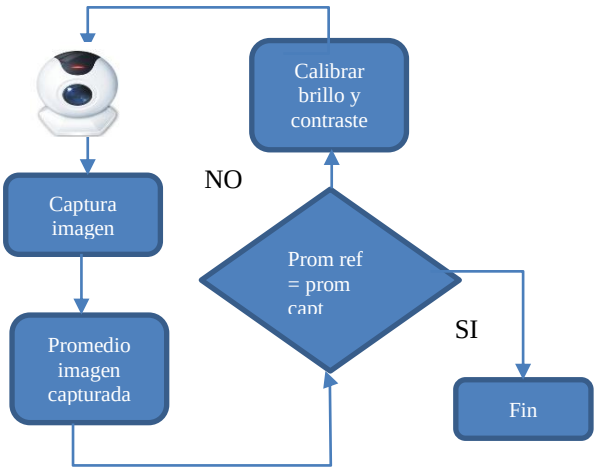

Figura 1 Arquitectura básica del sistema

El procedimiento se realiza de forma periódica para ajustar los parámetros cuando el sistema detecta un cambio de iluminación en la escena.

El sistema se encuentra divido en los siguientes módulos:

- Módulo de conexión
- Módulo de lectura y almacenamiento
- Módulo de procesamiento
- Módulo de control

El primer módulo permite la conexión con el driver de la cámara.

Para aprovechar el hardware del computador, a cada uno de los tres módulos siguientes le corresponde un hilo de ejecución lo cual permite que el sistema tenga mayor rendimiento y fiabilidad y dado que los hilos van a compartir información es necesaria la utilización de variables compartidas. [1]

El módulo de conexión permite la comunicación entre la cámara y sistema operativos, cuando la cámara es reconocida se

envía una señal al módulo de lectura y almacenamiento el cual se encarga de la captura, conversión y almacenamiento físico de los frames almacenados en memoria. Inicialmente contamos con un buffer que contiene los frames con codificación de imagen YUV debido a que este formato concentra la comprensión en el color preservando la luminosidad lo cual es conveniente, debido a que el ojo humano es más sensible a la luminosidad que al color, logrando reducir el tamaño de la captura , los frames ocuparan menos espacio y por lo tanto la transmisión es más rápida, además de facilitar los cálculos para obtener el promedio de la imagen una vez que la imágenes fueron transformadas a formadas a formato RBG, son enviadas al módulo de procesamiento para poder calcular el promedio la imagen capturada por el sistema y comparar este promedio con la imagen de referencia respectiva, el resultado es enviado al módulo de control donde se configuran los parámetros de la cámara.

Para la configuración de los parámetros de una cámara en este proyecto se consideran el brillo, contraste y balance de blancos.

Figura 2 Imágenes con variación de brillo y contraste

El brillo y el contraste son parámetros básicos, como se pueden observar en las figuras una modificación de cualquiera de ellos suele suponer un cambio visible en la imagen [3]. El parámetro brillo permite corregir problemas de iluminación en imágenes oscuras o demasiado claras, mientras que el contraste permite realzar los colores de una imagen, además el parámetro balance de blancos es un control de la cámara que sirve para ajustar el brillo de los colores básicos rojo, verde y azul (RGB) con el objeto de que la parte más brillante de la imagen aparezca como color blanco y la menos brillante como negro.

El ajuste de balance de blancos es necesario cuando existe una iluminación artificial que produce que uno de los componentes de color prevalezca sobre las otras.

4. Implementación del Software

El sistema fue implementado en LINUX. Para la creación y control de hilos se utilizó la librería pthread y para la comunicación con la cámara se utilizó la librería de V4L2.

Inicialmente el módulo de conexión establece la comunicación con la cámara, primero se crea un puntero que hace referencia al nombre del archivo de la cámara (dev/videoX) que se crea automáticamente cada vez que se conecta un dispositivo al computador, donde X es un índice que indica el número de cámaras más uno conectadas a la computadora, también se define el tipo de acceso que puede ser de lectura, escritura o ambos.

Debido a la variedad de dispositivos de video, se debe verificar que la cámara conectada sea compatible con el controlador de V4L2, para esto, se realiza una llamada al sistema con la función ioctl ( ) que recibe como parámetro el descriptor de archivos además se define el formato y el diseño de la imagen en memoria como el alto, ancho y el formato de pixel de la imagen, en nuestro proyecto utilizamos el formato de pixel YUV, una vez establecida la comunicación se inicia la captura de video, donde la imagen capturada en almacenada en un buffer en memoria.

Una vez que los frames se encuentren almacenados en el buffer, en el módulo de almacenamiento y lectura, se procede a extraer la imagen que se encuentran en formato YUV donde el parámetro Y representa la luminancia (es decir, información en blanco y negro), mientras que U y V representan la crominancia (es decir, información con respecto al color), luego se procede a realizar la transformación a formato de imagen RGB.

Para realizar la trasformación de YUYV a RGB es necesario tener en cuenta que cada cuatro bytes representan dos pixeles de RGB, es decir que a cada pixel le corresponde un valor de Y (luminancia) la combinación de U y V (crominancia). [4]

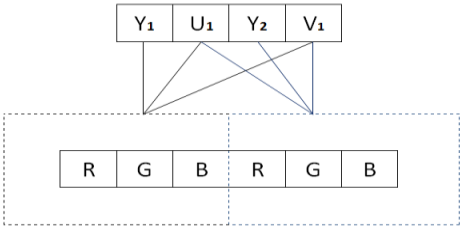

Figura 3 Formato YUV 4:2:2

Lo siguiente para culminar con la transformación, es multiplicar por un factor a los valores de luminancia y crominancia ese factor permite tener la proporción exacta de cantidad de luz que se necesita en cada color para representar un pixel en las siguientes ecuaciones se muestran las conversiones para obtener los diferentes canales de un pixel en RBG.

$$B = 1.164 * (y - 16) + 2.018 * (v - 128)$$

Ecuación 1 Conversión de espacio de color YUV al canal B

$$\boldsymbol{G = 1.164 * (y - 16) - 0.813 * (u - 128) - 0.391 * (v - 128)}$$

Ecuación 2 Conversiones de espacio de color YUV al canal G.

$$\mathrm{R} = 1.164 * (\mathrm{y} - 16) + 1.596 * (\mathrm{u} - 128)$$

Ecuación 3 Conversiones de espacio de color YUV al canal R.

A continuación en el módulo de procesamiento y control es donde se calcula el valor de brillo contraste y balances de blancos, para la calibración del parámetro brillo y contraste se procede a obtener el promedio de una de las imágenes ideales para posteriormente ir cambiando el parámetro brillo o contraste en la cámara hasta que el promedio de la imagen recientemente capturada sea igual al promedio de la imagen ideal. Para el proceso de calibración de balance de blancos, el proceso es similar, pues también es necesario obtener el promedio de la imagen pero de cada canal por separado e ir manipulando el parámetro de ganancia de rojos y azules hasta que se cumplan las siguientes igualdades:

$$\frac{\boldsymbol{Prom\ rojos\ img\ cap}}{\boldsymbol{Prom\ verdes\ img\ cap}} = \frac{\boldsymbol{Prom\ rojos\ img\ ref}}{\boldsymbol{Prom\ verdes\ img\ ref}}$$

$$\frac{\boldsymbol{Prom\ azules\ img\ cap}}{\boldsymbol{Prom\ verdes\ img\ cap}} = \frac{\boldsymbol{Prom\ azules\ img\ ref}}{\boldsymbol{Prom\ verdes\ img\ ref}}$$

Ecuación 4 Promedio de la imagen para la calibración de balances de blancos.

Para comunicación entre los procesos se utilizó el método de variables compartidas pero este método puede inducir errores de consistencia de datos, para evitarlos se puede utilizar diferentes mecanismos de sincronización como los semáforos que permiten restringir o conceder el acceso a recursos compartidos esto es importante para resolver problemas de sincronización y evitar condiciones de carrera "cuando dos o más procesos leen o escriben en un área compartida y el resultado depende de los instantes de ejecución de cada uno". [5]

Con este método logramos que exista una calibración automática de brillo, contraste y balance de blancos de una imagen en tiempo real, lo que evita un post procesamiento y por lo tanto nos permite tener un tiempo de respuesta mínimo.

5. Descripción de las pruebas

Se realizaron varios experimentos que permiten evaluar la eficiencia de la solución y la exactitud del algoritmo utilizado en el sistema. Para esto se tomaron muestras de las 8:00 de la mañana hasta las 18:30 de la noche y las condiciones ambientales del lugar en el momento que se realizaron los experimentos fueron nubladas y parcialmente nublado.

Para medir la eficiencia del sistema propuesto se diseñan pruebas que evalúan la eficiencia de ejecución del sistema y la robustez del algoritmo a cambios de iluminación.

Eficiencia de la solución: Se mide el tiempo que le toma al sistema procesar datos desde que detecta una variación en el ambiente hasta que calibra los parámetros de brillo y contraste. Esta medición se la realiza en dos formas:

1. Se compara el tiempo de ejecución al cambiar el número de procesadores asignados al sistema.
2. Se compara el tiempo de ejecución del sistema utilizando la librería OPENCV y el api de V4L2.

Exactitud del algoritmo de calibración: Se mide la similitud entre la imagen calibrada y la imagen de referencia, calculada en base a los promedios de las imágenes y la comparación de sus histogramas.

5.1 Eficiencia de la Solución

Experimento 1

Para este experimento las condiciones ambientales de la captura de la imagen de referencia fueron las mismas utilizadas en la calibración del algoritmo. Del mismo modo, se usó la misma imagen de referencia para la ejecución del algoritmo con diferentes números de procesadores, esto se realizó para obtener la diferencia entre los distintos tiempos de ejecución con uno o varios procesadores.

Experimento 2

Para este experimento se desarrolló el sistema con la librería de OPENCV y el api de V4L2 estos fueron ejecutados bajo las mismas condiciones ambientales y escenarios utilizando la misma imagen de referencia. El objetivo de

este experimento es comparar los tiempos de ejecución del programa usando diferentes librerías.

5.2 Exactitud del Algoritmo de Calibración

En esta sección se procede a evaluar la exactitud del algoritmo a los diferentes cambios de luz ambiental, por facilidad para el desarrollo de este experimento se utilizó la imagen de referencia y las imágenes resultantes procesadas en el experimento 2 con el api de V4l2. Posteriormente se obtuvo el histograma de cada una de las imágenes y se utilizó la métrica de χ^2 (chi cuadrada) para saber si existía o no relación entre el histograma de la imagen de referencia y el histograma de cada imagen resultante. La distribución Chi cuadrado permite calcular si los resultados estadísticos de un experimento se alejan significativamente o no de los resultados esperados del modelo teórico.

6. Resultados

A continuación se presentan los resultados obtenidos de los experimentos realizados para medir la eficiencia d la solución y la robustez del algoritmo a cambios de iluminación mencionados en el punto 5:

6.1 Eficiencia de la Solución

Experimento 1

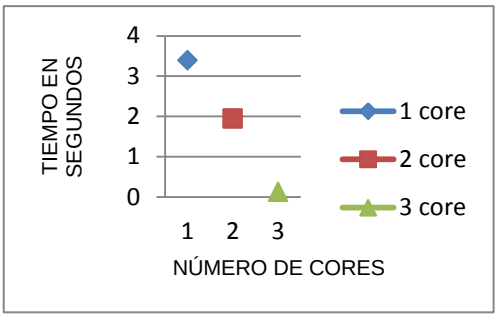

Figura 4 Tiempo de procesamiento vs número de procesadores del CPU

Al asignarle los tres hilos (algorithm, camera, control_camara) a un sólo procesador el tiempo de ejecución es mayor debido a que todo el trabajo obviamente es gestionado por un solo CPU y por lo tanto, los bloqueos y periodos de espera son más frecuentes debido a las posibles interrupciones provenientes de otros procesos que requieren el uso del mismo CPU.

Por el contrario, si a cada hilo le asignamos un procesador diferente, el tiempo de ejecución se reduce 96% lo cual es lógico porque el trabajo se encuentra distribuido en varios procesadores. Adicionalmente, el hecho de que cada hilo se ejecute en un procesador ayuda a incrementar los aciertos de caché y eliminar el consumo de la infraestructura de cambio entre hilos.

Experimento 2

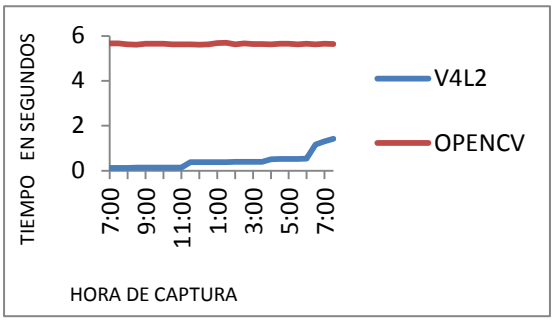

Figura 5 tiempos de procesamiento en V4L2 vs OPENCV

En la Figura 5 se puede apreciar que utilizando el programa A (librería V4L2) los tiempos de respuesta son menores a los tiempos de respuesta del programa B (librería OpenCV), esto se debe a que la librería de V4L2 trabaja en un lenguaje de bajo nivel, no utiliza librerías externas y permite la comunicación directa con el kernel de Linux.

6.2 Exactitud del Algoritmo de Calibración

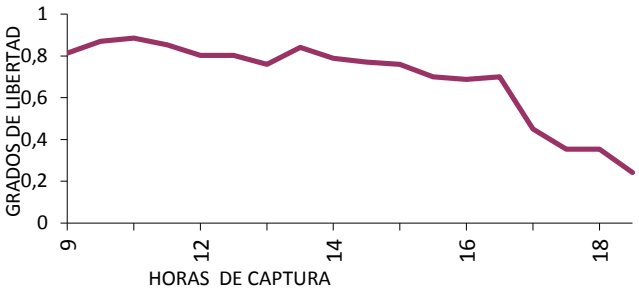

Figura 6 Grados de libertad entre la imagen de referencia y cada imagen resultante.

En la Figura 6 se graficaron los resultados obtenidos por chi cuadrada, entre más alto sea el valor obtenida por la chi cuadrada mayor relación existe entre el histograma perteneciente a la imagen de referencia y el histograma de las imágenes resultantes del algoritmo de calibración de la cámara, por lo que se puede

concluir que en condiciones normales de iluminación se cumple que a medida de que la luz disminuye el algoritmo es menos exacto con respecto a la imagen de referencia.

En estas pruebas se utilizaron los histogramas debido a que permite representar la cantidad de pixeles que aparecen en una imagen con cada determinado valor de luminosidad, lo cual también nos indica que el valor promedio de la imagen ya que esta es la suma de valores de luminosidad de cada pixel divido para el número de pixeles, lo cual indica si dos imágenes poseen el mismo valor promedio no exactamente son imágenes iguales, pero sí que posean el mismo valor de luminosidad.

7. Conclusiones

Las condiciones ambientales afectan notablemente el proceso de calibración de la cámara a un punto donde la calibración ya no es posible.

La ejecución del sistema se realiza en promedio cinco segundos más rápido que si utilizáramos una librería externa como OpenCV.

La programación concurrente permite reducir los tiempos de ejecución sobre plataformas multiprocesadoras.

8. Recomendaciones

Para mayor aprovechamiento del sistema tener en cuenta las características del CPU ya que el sistema se encuentra diseñado de tal manera que permite aprovechar la afinidad del procesador.

Realizar las capturas de las imágenes de referencia en las condiciones ambientales adecuadas.

Asegurarse que no exista ninguna fuente cercana que produzca alguna clase de ruido electrónico en el sensor de la cámara.

Revisar si el dispositivo de captura permite la configuración de los parámetros de brillo, contraste y balance de blancos.

9. Agradecimientos

A Dios que me permite alcanzar este logro y compartirlo con mi familia. A mis padres que han sido mi apoyo durante todo este tiempo, por los valores que me han inculcado pero sobre todo por ser el mejor ejemplo de vida a seguir.

10. Referencias

[1] P. B. G. G. G. Abraham Silberschatz, Fundamentos de Sistemas Operativos, Madrid: McGraw-Hill/Interamericana de España, S.A.U, 2006, p. 3.

[2] H. Nanda, «Practical calibrations for a real-time digital omnidirectional camera,» Maryland: University of Maryland..

[3] P. porta, «quesabesde,» [En línea]. Available: http://www.quesabesde.com/camdig/articulos.asp?articulo=153. [Último acceso: 08 08 2013].

[4] M. H. S. H. V. M. R. B. Dirks, Video fro Linux two API Specification, copyright, 2008.

[5] lrmdavid@exa.unne.edu.ar, «SISTEMAS OPERATIVOS,» [En línea]. Available: http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO0.htm. [Último acceso: 20 05 2013].