Efecto de ojos rojos

Laura del Pino Díaz – Máster Oficial en Ciencia de Datos e Ingeniería de Computadores

Visión por Computador

Contenido

**No se encontraron entradas de tabla de contenido.**

# El efecto de ojos rojos

El efecto ojos rojos sobre una imagen se produce cuando el flash de la cámara, se refleja en el fondo del ojo, devolviendo ese color rojo. Esto se suele producir porque la persona a la que se fotografía tiene la pupila dilatada porque se encuentra en un ambiente oscuro en el que necesita que entre más luz en el ojo para poder ver y de repente se encuentra con un punto muy luminoso que es el flash de la cámara. Sin tiempo de reacción para contraer la pupila la luz del flash entra en el ojo y se refleja en el fondo del mismo, generando el efecto de ojos rojos.

El objetivo de este trabajo es corregir el efecto de ojos rojos de imágenes estáticas. Para ello tenemos que detectar la zona de los ojos y modificar el color de la pupila. También nos interesa saber la bondad de nuestro algoritmo de corrección por lo que crearemos una métrica que nos ayude a determinar la bondad del algoritmo.

# El bajo nivel del efecto de ojos rojos

Antes de meternos con el desarrollo del algoritmo queremos conocer que es lo que pasa a nivel de representación en el espacio de color con los píxeles que determinamos que son efecto de ojos rojos.

Para ello utilizamos Matlab para cargar la imagen y la herramienta imtool para seleccionar la sección de la imagen que contiene un ojo rojo utilizando la herramienta *crop image* de la barra de herramientas.

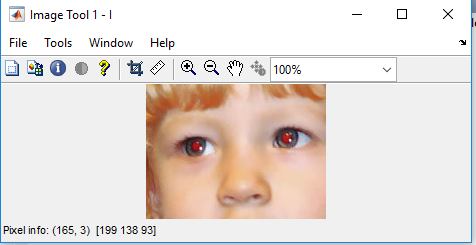


Ilustración 1: imagen visualizándose en imtool dentro de Matlab

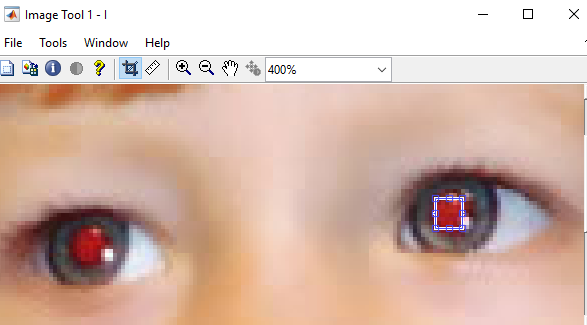


Ilustración 2: Selección de ojos rojos usando la herramientoa crop.

Una vez tenemos la selección la exportamos al espacio de trabajo. Se le aplican correcciones para cada una de las tres componentes RGB estén en una columna y cada uno de los píxeles en una fila.

Tras ello se calculan los valores máximos y mínimos que tenemos en esta zona, así como el intervalo calculado sobre ellos y el porcentaje dentro de la escala de representación que representa este intervalo.

|  |  |  |  |
| --- | --- | --- | --- |
|  | R | G | B |
| Mínimo | 102 | 12 | 15 |
| Máximo | 219 | 89 | 87 |
| Longitud Intervalo | 117 | 77 | 72 |
| Porcentaje sobre la escala de representación. | 45.88% | 30.2% | 28.24% |

Debido a los altos porcentajes sobre la escala de representación, se puede llegar a la conclusión de que el espacio RGB no es el ideal para discriminar rápidamente los valores correspondientes a esta zona, por lo que vamos a repetir el experimento en el espacio HSV y obtenemos la siguiente tabla.

|  |  |  |  |
| --- | --- | --- | --- |
|  | H | S | V |
| Mínimo | 0.0025 | 0.6027 | 0.4 |
| Máximo | 0.9987 | 0.9153 | 0.8588 |
| Longitud Intervalo | 0.0038 | 0.3126 | 0.4588 |
| Porcentaje sobre la escala de representación. | 0.38% | 31.26% | 45.88% |

En este caso, el porcentaje de posibles píxeles que podrían ser ojos rojos se reduce al tener solamente un 0,38% de probabilidades de tener el tono de color adecuado.

La pregunta que tenemos que responder en este momento es: ¿Todos los ojos rojos pertenecen a este intervalo?

Ante un experimento en el que se carga otra imagen y se ponen a 0 todas aquellos píxeles que no pertenecen al intervalo para cada una de los componentes y se cuentan todos aquellos componentes distintos de 0 obtenemos que 93 píxeles no son 0 y además la siguiente imagen.

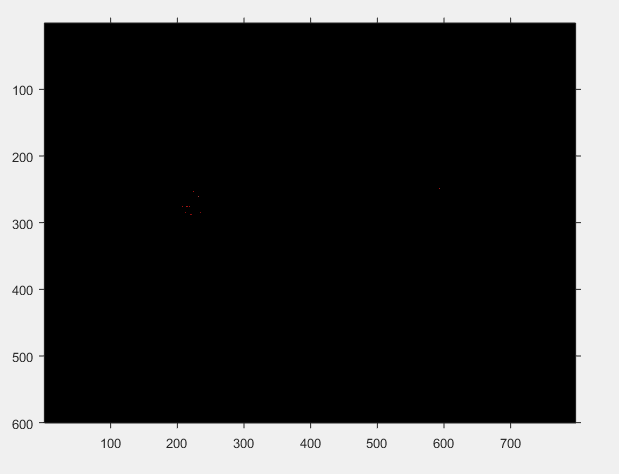


Ilustración 3: Píxeles que no pertenecen al intervalo.

En la imagen aparecen un pequeño conjunto de puntos dispersos de color rojo. Lo ideal es que fuese una superficie más o menos continua. Por lo que se puede deducir que el intervalo es demasiado pequeño. Para ampliarlo planteamos el siguiente experimento:

Cargar un número de imágenes de los que extraemos los ojos y al igual que antes los pasamos al espacio de color hsv para determinar el mínimo y el máximo de cada una de las componentes.

Separando a mano cada uno de los ojos de las imágenes de prueba desde ojo1.jpg hasta ojo5.jpg obtenemos una tabla con los siguientes datos:

|  |  |  |  |
| --- | --- | --- | --- |
|  | H | S | V |
| Mínimo | 0 | 0.039 | 0.1412 |
| Máximo | 0.9993 | 1 | 1 |
| Longitud Intervalo | 0.007 | 0.9610 | 0.8588 |
| Porcentaje sobre la escala de representación. | 0.07% | 96.10% | 85.88% |

Ante estos datos sacamos dos conclusiones:

1. Los valores de Saturación (S) e iluminancia (V) no nos sirven para discriminar los colores que nos interesan, puesto que toman prácticamente todo el conjunto de valores posibles.
2. Dado que el valor del tono es el único que no es lineal en este espacio de color, sino que es radial, tanto en este experimento como en el anterior los límites del intervalo se están calculando mal.

A la hora de probar a obtener los límites a partir de imágenes recortadas a mano obtenemos valores dispersos por todos los tonos por lo que se considera este experimento inválido y se procede a determinar que los ojos rojos serán aquellos colores comprendidos entre el valor 0.9 y 0.1 del componte HUE del espacio HSV y aquellos que tengan una alta saturación, es decir que tengan una componente S superior a 0.85.

Siguiendo esta idea obtenemos una serie de ejemplos reproducibles mediante el script *lowlevelRedEyeExperiment3.m*  en el que vemos que existen casos en los que sí que es cierto que se detecta bien los ojos rojos y otros no, como por ejemplo en los que la saturación no es alta.

El anterior mencionado script hace uso de una función externa llamada *getMask*  que devuelve una imagen en blanco y negro donde los píxeles en blanco son aquellos que cumplen la condición de Hue entre 0.9 y 0.1 y la Saturación a más de 0.85. Dado que en algunas imágenes los ojos rojos no se detectan a causa de que no cumplen la componente de saturación se creará una segunda versión de *getMask* para replicar el experimento y comprobar si detecta más casos de ojos rojos.

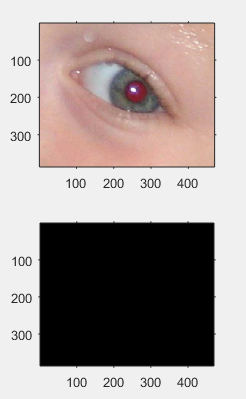


Ilustración 4: Ejemplo de no detección de ojos rojos

Esta nueva versión, genera una alta tasa de falsos positivos, como se muestra a continuación:

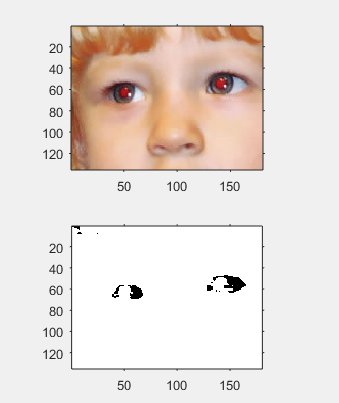


Ilustración 5: Ejemplo de alta tasa de falsos positivos

Para evitar este tipo de casos, se retrocede en los pasos dados recuperando la discriminación por saturación y se reduce esta. Probando con valores menores de saturación tenemos algunos resultados mejores.

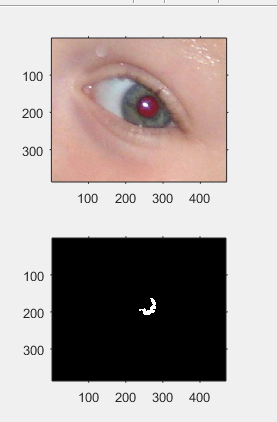


Ilustración 6: Imagen original con la máscara obtenida.

Pero también dan más falsos positivos:

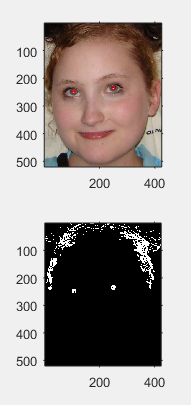


Ilustración 7: Ejemplo de aumento de falsos positivos cuando el umbral de saturación es 0.7

De esta serie de experimentos concluimos que los ojos rojos están representados con píxeles que están representados con un valor de tono entre 0.9 y 0.1 de Hue/tono, con un valor de saturación superior a 0.7. Pero esto genera falsos positivos y además algunos de los píxeles de los ojos rojos no se detectan, debido a que tienen una saturación menor.

# Bibliografía

[1] A GMM Based Automated Red Eye Detection Algoritm – Mithun Uliyar,Krishna A.G.,P.S.S.B.K.Gupta,J.P.Pai – IEEEXplore 2009