Balance de blancos con más de una iluminación

Marc Solà Ramos

Resumen— Este proyecto es una herramienta que tiene como objetivo principal conseguir una aplicación local de algoritmos de detección de un solo iluminante con la intención de detectar múltiples iluminantes en una misma imagen donde la luz de la escena es una mezcla de luces que provienen de orígenes distintos, como por ejemplo un habitación iluminada por una bombilla y a la vez por luz proveniente de la calle que entra por la ventana. Para la aplicación local de los algoritmos de un solo iluminante el proyecto presenta 2 metodologías distintas. El proyecto base que se usa para hacer pruebas de dichas metodologías para la detección de múltiples iluminantes usa dos funciones Gaussianas que son conocidas dentro de la comunidad científica por emular el comportamiento de las neuronas encontradas en las capas V1 y V4 del sistema visual humano. Aparte de las dos metodologías para la detección de múltiples iluminantes, este proyecto presenta 3 metodologías distintas para la corección de imagen y propone una mejora para el proyecto base que ayuda a corregir la pérdida de brillo que había tras corregir la iluminación de la imagen tratada.

Palabras clave— Neurona, V1, V4, region, length, width, incX, incY, Gaussian, Centre, Surround, k-means, cellArray, iluminante, multiples, local.

Abstract— This Project is a tool intended for the local application of single illuminant detection algorithm with the goal to detect multiple illuminants in the same image in which the light comes from different sources, like for instance a room i illuminated at the same time by a light bulb and also light coming from the outside through a window. This Project features two different methods to approach the local application of single illuminant detection algorithms. The project provided to test said methods in order to get multiple illuminant detections is based on two Gaussian functions which are known to fit the behaviour of neurons found in the V1 and V4 layers of the human visual system. Besides the two methods to get multiple illuminant detections the project also features three illuminant correction methods and it has a proposal to improve the provided project, which had a loss in the brightness of the final colour constant image after the illuminant correction was applied.

Index Terms— Neuron, V1, V4, region, length, width, incX, incY, Gaussian, Centre, Surround, k-means, cellArray, illuminant, multiple, local.

1 Introducció

El color de la iluminación es un aspecto muy importante en fotografía. La luz que ilumina la escena de una fotografía puede venir de diferentes fuentes y en muchos casos es así. En un paisaje en el exterior por ejemplo mayormente vendrá provenida del sol, pero también puede provenir del reflejo en otros objetos o incluso retransmitirse des del mismo cielo. En cotextos de interior, por ejemplo una habitación, la luz proviene de las bombillas, que a su vez normalmente están rodeadas por un cristal de la lámpara que las contiene o de focos en el caso de un estudio de fotografía. En todos los escenarios descritos anteriormente también se puede encontrar luz que llega de forma indirecta, como es el caso de las sombras, lo cual es otro factor claro a tener en cuenta.

El color, que es el problema a tratar aquí, puede llegar a ser extremadamente distinto. En el escenario exterior del paisaje, la luz que llega proviene del cielo y tiene un tinte azulado debido a qué el cielo hace de filtro, ya qué se sabe que la luz real que desprende el sol de forma directa tiene un tinte rojizo-amarillento dependiendo de la hora del día. En escenas interiores esto se complica aún más por la presencia de diferentes tipos de lámparas y focos, que a su vez pueden combinarse con la luz que viene del exterior.

Por fortuna el sistema visual humano tiene integrado un algoritmo que hace el trabajo de filtrar las diferentes iluminaciones y ayuda a recuperar lo que percibimos como los colore reales de los objetos en una escena. Dicha propiedad se llama "constancia de color". Desde los comienzos de la fotografía, imitar dicha propiedad en la medida de lo posible para poderla incorporar en sus productos ha sido el gran empeño de los fabricantes de este mundo. El método usado por los fabricantes es el conocido "balance de blancos". Aún así, este método suele fallar en casos reales cuando en la escena la iluminación es una combinación de luz de dos o más fuentes de colores distintos.

- *E-mail de contacto: marcsr03@gmail.com*
- *Mención realitzada: Computación*
- *Trabajo tutoritzado por: C. Alejandro Parraga (Centro de Visión por Computador)*
- *Curso 2017/18*

La solución propuesta en este proyecto para resolver el problema del "balance de blancos"es el uso del algoritmo de detección de un solo iluminante de [5] A. Akbarinia y C. A. Parraga basado en la emulación de partes del sistema visual humano y dos metodologías de aplicación local diferentes de este para obtener muestras de un solo iluminante en cada región local de una misma imagen. El funcionamiento del algoritmo de A. Akbarinia se explica en detalle en el apartado de detección de un solo iluminante. Las metodologías de aplicación se explican con detalle en el apartado de detección de multi iluminante.

Posteriormente usando las detecciones prévias conjuntamente con el algoritmo [9] K-means se obtiene una detección de tantos iluminantes como clústers tiene k-means, aunque en este proyecto la idea es detectar dos iluminantes distintos.

Finalmente hay tres propuestas de corrección de la iluminación de la imagen a partir de los iluminantes calculados que se explican en detalle en el apartado de corrección de la iluminación de imagen.

2 Estado del arte

El problema del balance de blancos en fotografía es algo que preocupa a todos los fabricantes de cámaras digitales y de móviles porque para conseguir una mejor calidad en las fotografías tomadas es necesario en muchas ocasiones aplicar una corrección a dicha foto, la cual consiste en sustraer el color de la iluminación global de la foto con el objetivo de obtener una imagen nueva con los colores de tal forma que se vean como si estuvieran iluminadas por una luz neutra, es decir, una luz blanca o prácticamente blanca.

Este proceso, que se intenta perfeccionar tanto como se puede consta de dos grandes partes, que son la detección de los diferentes iluminantes que se detectan en la fotografía tomada y por último la corrección de la foto respecto a estos. Hay diferentes formas de atacar estas dos tareas.

En el caso de [6], D. Cheng y su equipo intentan hacer una detección de los iluminantes de una imagen dividiéndola en subregiones y aplicando árboles de regresión que usan de inputs especificaciones de la cámara que se usó para tomar las fotos del dataset. El problema en el trabajo de D. Cheng surge a la hora de hacer la corrección de la luz ya que este método no es efectivo para hacer una corrección de imagen con variación espacial.

En el trabajo de [5] A. Akbarinia por ejemplo se ha usado una aproximación al problema de detección que intenta emular el funcionamiento de las neuronas de las capas V1 y V4 encontradas en el cerebro de los seres vivos usando diferentes gaussianas [10] en función de su centro de contraste con los alrededores. Aunque en este caso se obtuvieron grandes resultados para datasets distintos en la detección y corrección de un single illuminant, los reultados para el caso de más de un iluminante no fueron nada favorables.

Otra forma de atacar el problema se puede ver en el caso de [11], S. Bianco y sus colegas para la detección se basan en el uso de una red neuronal que a base de convoluciones produce estimaciones locales del iluminante. En la siguiente fase se recogen todo los diferentes iluminantes detectados y finalmente se refinan con el uso de una agregación no lineal, lo cual proporciona una estimación global en caso de tener un único iluminante. Para la corrección en este caso usan el modelo de Von Kries.

Como se ha visto en los casos comentados, la estimación de los iluminantes en una escena suele ser la parte donde la mayoría de algoritmos son robustos, ya qué siempre y cuando la detección de almenos un iluminante se haga de forma correcta, extender la capacidad del algoritmo para una estimación de múltiples iluminantes consiste en encontrar una metodología para aplicar el algoritmo de forma local múltiples veces.

Donde parece aparecer el problema normalmente es a la hora de encontrar en algunos casos metodologías para la parte de la corrección global de la imagen respecto los iluminantes detectados.

3 Motivación

La principal motivación a la hora de escoger y realizar este proyecto ha sido poder colaborar en la mejora de algoritmos que tratan de solucionar el problema de balance de blancos de la fotografía digital usada tanto en cámaras y móviles, lo cual lo hace un problema bastante relevante debido al alto uso de estas tecnologías en la actualidad.

Además también la oportunidad de poder trabajar de forma parecida a como trabajan la gente que se dedica a la investigación de visión por computador, para poder experimentar enfoques de trabajo nuevos.

Por último, el saber que el trabajo estaba programado en MATLAB abría posibilidades a mejorar el entendimiento personal de este lenguaje, muy utilizado por su gran uso de las matrices y de sus indexaciones.

4 Objetivos

El objetivo principal de este proyecto es explorar soluciones computacionales al problema del balance de blancos en fotografía para recuperar los colores originales de los objetos en escenas donde la iluminación proviene de una combinación de fuentes luz, ya sean artificiales o naturales (por ejemplo, una luz fluorescente y una ventana en

un día soleado). Después de una familiarización con el problema y el material proporcionado se pueden deducir las siguientes tareas:

- Pensar en diferentes metodologías para mejorar la aplicación de algoritmos de detección de un solo iluminante para conseguir una detección de múltiples iluminantes, en especial el algoritmo ColourConstancySurroundModulation proporcionado por A. Akbarinia [5].

- Aplicar esas metodologias usando un datset en el cual se ha verificado que las escenes contienen más de un iluminante.

- Comrpobar la eficiencia de las metodologías usando las métricas recovery angular error y reproduction angular error conjuntamente con el groundtruth del dataset.

- Pensar en diferentes formas de aplicar una corrección a la imagen a partir de los diferentes iluminantes detectados con el algoritmo.

5 Planificación

Tabla 1. Planificación del proyecto por horas.

Tareas	**Horas**
Búsqueda y lectura de información	50
Familiarización con el código del proyecto base	30
Tutorías y especificación del proyecto	20
Implementación de las dos metodologías en la función GetSpatialRegions	35
Implementación de cambios de CalculateLuminanceMulti para adaptarla a la funciones de GetSpatialRegions y GetMultiIlluminants	20
Implementación de MultiIlluminantCorrection	15
Implementación de AngularErrorSetUp1 y AngularErrorSetUp2	5
Implementación de PixelAngularError	5
Implementación de ReproductionAngularError	15
Implementación CalculateAngularErrorV1 y CalculateAngularErrorV2	20
Realización de pruebas	10
Redacción de informes	40
Preparación de la presentación	5

6 Metodologia y desarrollo

Para llevar a cabo el proyecto se ha elegido el lenguaje de programación MATLAB debido a distintas razones:

- MATLAB es lenguaje en el que está programado el proyecto de A. Akbarinia [5], que es el proyecto de base que se me ha proporcionado. Otros proyectos que se pueden usar para la detección de illuminants como el de D. Cheng [6] también han sido programados con MATLAB.

- MATLAB es considerado por muchos un gran lenguaje de programación para todo lo relacionado con el tratamiento de imágenes a nivel computacional debido a qué cualquier imagen se acaba traduciendo en una estructura matricial y MATLAB gestiona dichas estructuras de forma espléndida tanto en su creación como a la hora de hacer accesos a ellas de muchas formas distintas y útiles en cada caso respectivamente.

Como IDE se ha utilizado MATLAB R2015a ya qué en años anteriores fue la IDE que se usó en otras asignaturas de tratamiento de imágenes y también porque tras ejecutar el proyecto base no dio ningún problema por antigüedad de versión. Los Datasets principales que se usan para las pruebas del proyecto son [7] y [8].

En los siguientes apartados se explicará los pasos que se siguen en el proceso de la detección de iluminantes y corrección de imágenes.

6.1 Detección de un solo iluminante usando el proyecto proporcionado

La idea base de este proyecto [5] es tomar una imagen con almenos una fuente de luz y usando principios biológicos sobre como percibe los colores el sistema visual de animales vivos [2], conseguir extraer cual es el color predominante de dicha fuente de luz respecto a la escena de la imagen.

En el caso de un solo iluminante el modelo intenta simular el comportamiento de los receptive fields (RF) en V1[2](p.19) usando una convolución con una matriz máscara que sale de hacer una diferencia de Gaussianas (DoG). Las Gaussianas [10] en cuestión son las conocidas como "Narrower Gaussian" y "Broader Gaussian", ya que se conoce que simulan bien las respuestas neuronales. La primera de ellas para simular dichos receptores en centre y la segunda para la parte de surround.

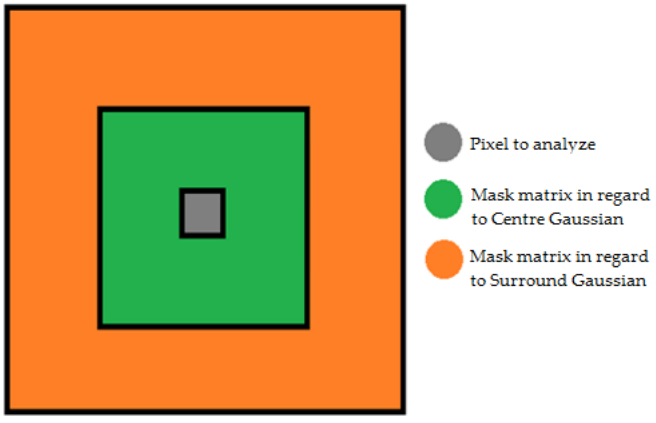

Fig.1. Máscaras aplicadas a la hora de computar un píxel respecto a "Centre" y a "Surround".

En la Fig.1 se puede observar siguiendo la leyenda que por cada píxel a analizar hay una máscara para el centre y una para surround. Las máscaras se toman siempre en el centro del píxel en cuestión y tiene dimensiones n x m según convenga. Para computar cada píxel, a la hora de hacer la convolución y aplicar la máscara, esta se aplica de forma simultánea a todos los canales del píxel por separado (R,G,B). Previo a la convolución es necesario preparar las máscaras. Para ello primero se calcula el contraste local usando desviación estándard y se determina el kernel y su tamaño a usar en el píxel en cuestión.

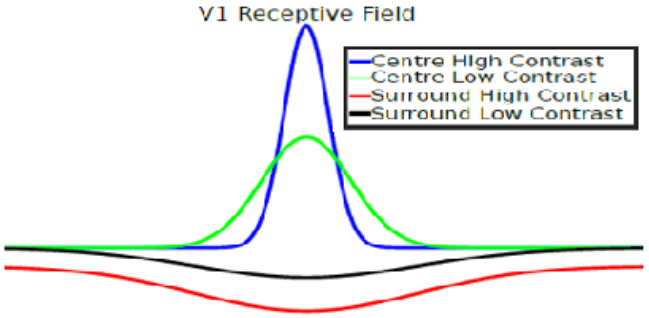

Fig.2. Curvas de las Gaussianas respecto del "Centre" y del "Surround".

Respecto a los RF de centre, para calcular su respuesta (CR) se hace la convolución usando la narrower Gaussian con valores de anchura dentro del intervalo [σ,2 σ] para el Gaussian kernel. Las dimensiones de este son inversamente proporcionales al contraste local calculado en cada dirección (x,y). Debido a esto para hacer la computación en el caso ideal se generaría un narrower Gaussian kernel para cada píxel, pero el coste computacional sería muy alto. Para evitar dicho problema se hace una quatización uniforme, que consiste en establecer un rango acotado entre los dos valores extremo obtenidos después de calcular el contraste local de todos los píxeles de la imagen y dividiendo el rango obtenido por un número arbitrario. Esto genera diferentes intervalos de rangos a los que se les denomina niveles (L). Supongamos por ejemplo que este intervalo es [0, 1]. En la Fig.2 se pueden observar 2 gaussianas distintas tanto para centre como para surround, es decir hay 2 levels en ese caso concreto, que darán lugar a los intervalos [0.0, 0.5] con una gaussiana de σ y (0.5, 1.0] con una gaussiana de 2 σ .

Respecto a los RF de surround para calcular su respuesta (SR) se hace una convolución de la imagen original sobre cada canal usando el broader Gaussian kernel. La diferencia principal respecto a la CR es que el size escogido (5 σ) a la hora de generar la máscara es el mismo para ambos ejes sin importar el contraste local, es decir n x n. Esta elección es debida al hecho de qué la variaciones del contraste local son mucho más pequeñas que los RF de centre en los diferentes niveles.

La respuesta global (RR) en V1 se obtiene de la suma de las dos anteriores y multiplicando cada una por un peso específico. Estos pesos están modelados de tal forma que consideran el hecho de qué la fuerza de CR y la supresión causada por surround dependen del contraste y de las orientaciones relativas del estímulo tanto de centre como de surround. Tanto el peso de centre (λ) como el de surround (k) son inversamente proporcionales al contraste orientado respecto a centre y a surround respectivamente. En casos en los que el centro está expuesto a bajo contraste o cuando el estímulo central y el de surround son ortogonales el uno respecto al otro, la supresión se convierte en facilitación, es decir k tomaría valores positivos, lo cual transformaría el modelo de una DoG a una SoG (Suma de Gaussianas).

Una vez acabado esta parte del proceso se intenta modelar ciertas características de la capa V4[2](p.25), dónde los científicos creen que se produce la parte del procesado final de colour constancy. Para hacerlo se coge un porcentaje de las neuronas dominantes del proceso de la capa V1 y se hace un pooling con ellas para formar el RF de la capa V4 que es 9 veces mayor que el de la capa V1. A nivel hipotético este pooling lo que hace es que cuando un RF de una neurona en V4 experimenta un contraste alto, un pequeño porcentaje de neuronas V1 activan y ese porcentaje es el que forma parte del RF de una neurona en V4. A este porcentaje se le denomina "winner-takes-all" y se hace por cada canal (R,G,B).

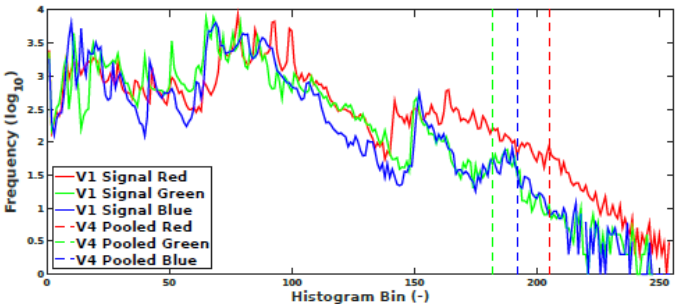

Fig.3. Histograma de la señal RR captada por los RF de V1 en los 3 canales de color (R,G,B). Cada una de las líneas discontínuas marca el Bc (threshold) de casa canal a partir del cual hay las neuronas más activadas ("winner-takes-all") que se usan para el pooling de las neuronas de V4.

Para implementar "winner-takes-all" se calcula la media del contraste local de todos los inputs a V4 dado un canal de color usando la desviación estándar de los píxels de RRc computados usando el RF medio en V4. A esta media se la denomina Pc y es el porcentaje sobre 1 que se usa para determinar el threshold Bc. En la Fig.3 se puede ver un histograma de una imagen donde se ve la distribucuión de las diferentes señales de frecuencia captadas en V1 para cada canal en escala $\log_{10}$. Usando Pc sobre el total de RR en cada canal repectivamente en el histograma, se determina en que bin se encuentra Bc, que en el histograma correspondería a las líneas discontínuas. Desde Bc incluido hasta el último bin del histograma se hace el pooling para V4. Finalmente sumando en cada canal de color respectivamente las contribuciones de los bins dentro del pooling de V4 se obtiene el color estimado del iluminante.

6.2 Detección de múltiples iluminantes

En esta sección se explica en detalle los diferentes pasos que se han seguido en la detección de múltiples iluminantes de una imagen.

Ahora se explicarán las dos metodologías de aplicación local del algoritmo de detección de un solo iluminane. Cabe recalcalcar que dichas metodologías son fácilmente adaptables a cualquier otro algoritmo de detección de un solo iluminante, siempre y cuando se adapten bien las estructuras de datos necesarias.

6.2.1 Metodología 1

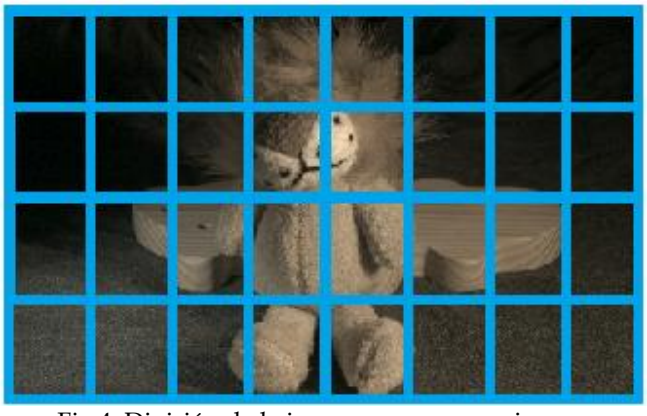

Fig.4. División de la imagen en n x m regiones.

Esta metodología, como se puede apreciar en la Fig.4, consiste en dividir la imagen en celdas por fila (n) x regiones por columna (m). Las dimensiones de dichas regiones se obtienen a partir de las Eq.1 y la Eq.2.

$$region\ width = floor(image\ width/n) \quad (1)$$
$$region\ length = floor(image\ length/m) \quad (2)$$

El inicio de cada región se considera el canto izquierdo superior de la región. Para cada región estos datos se guardan en la estructura cellArray conjuntamente con region width y region length. Usar la operación floor en Eq.1 y Eq.2 es necesario porque ambas se usan posteriormente para hacer accseos a la estructura y solo se pueden tener índices enteros. En el caso que region width o region length no tuvieran división exacta se hace un reshape de la imagen a tratar y de la matriz ModelResponse creando una matriz con las nuevas dimensiones que salen a partir de Eq.3 y Eq.4 porque hay píxeles que no se usan en la detección de iluminantes y por lo tanto no son necesarios.

$$new\ rows = region\ width * n \quad (3)$$
$$new\ cols = region\ length * m \quad (4)$$

Una vez acabados los reshape se puede proceder a la aplicación de la detección de un solo iluminante en cada región haciendo accesos a las posiciones guardades en cellArray.

6.2.2 Metodología 2

Esta metodologia, como se puede apreciar en la Fig.5, consiste en generar una celda que se desplaza por la imagen para hacer las detecciones locales de un iluminante. Como en el caso de la metodología 1 las dimensiones de la celda vienen dadas por las Eq.1 y la Eq.2 y también se hace un reshape de la imagen si es necesario. El cambio principal entre la metodología 2 y la 1 es la aparición de incX y incY, que se calculan a partir de Eq.5 y Eq.6

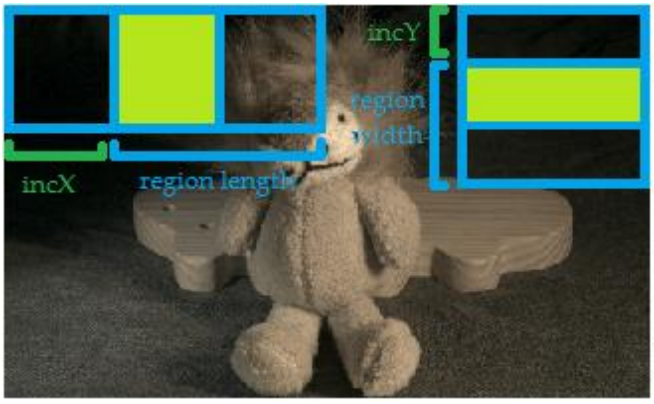

Fig. 5. Celda de dimensiones region length x region width que se desplaza a lo largo de la imagen.

$$incX = region\ length/PartitionJTmp \quad (5)$$
$$incY = region\ width/PartitionITmp \quad (6)$$

donde PartitionJTmp y PartitionITmp son respectivamente el número más alto en el intervalo [1, 13] por el cual region length y region width pueden ser divididos de forma entera., es decir se hace la operación mod(region length, número de iteracion bucle) y si da 0 pues se guarda en la variable de PartitionJTmp el número de iteracion bucle. Se hace por cada iteración de la misma forma también con region width y PartitionITmp. La decisión del intervalo viene del hecho de qué para asegurar que incX y incY den un incremento pequeño y entero es bueno usar las medidas de la celda y el hecho de qué en ese intervalo hay los primeros 6 números primos para poder encontrar número alto por el cual esa dimension sea divisible y así aumentar la probabilidad de conseguir unos incrementos pequeños para poder generar más detecciones. El hecho de qué 1 forme parte del intervalo es para cubrir el peor caso en el que region width o region length no sean divisibles por un número más grande que 1, lo cual significaria que el desplazamiento en ese eje sería igual a la dimensión de la celda en ese eje. El intervalo de [1, 13] se puede extender a los siguientes números primos si así se desea y se dispone de una buena capacidad de computo, però siempre manteniendo el extremo derecho del intervalo por debajo del valor de la dimensión más pequeña de la celda.

Del mismo modo que en metodología 1 las posiciones que tomarà la celda se guardan conjuntamente con sus dimensiones, incX y incY en la estructura de cellArray. El

hecho de qué incX y incY normalmente sean distintos al tamaño de celda respecto a ese eje produce el fenómeno que se puede observar en Fig.5 en las zonas pintades de verde amarillento, en el cual hay píxeles de la imagen que a la hora de hacer las detecciones de un solo iluminante por región son utilizados en múltiples detecciones. En el apartado de resultados veremos como ha afectado a eso a la detección de múltiples iluminantes.

6.2.3 Obtención de los iluminantes

Las detecciones hechas usando cualquiera de las dos metodologías y el algoritmo de detección de un solo iluminante se guardan en la estructura luminance por componentes (R,G,B). Es decir, el contenido de luminance se puede entender como una distribución de puntos en el espacio de color de RGB [4]. A dicha distribución de puntos se le aplica el algoritmo de k-means de MATLAB con dos clústers ya que en este proyecto solo se quieren encontrar dos iluminantes. Hay que tener en cuenta que la versión de k-means de MATLAB siempre inicializa los centroides de forma aleatoria. Una vez finalizada la ejecución se toma la posición final de los dos centroides como los iluminantes detectados y se guarda en la estructura de illuminants por componentes (R,G,B).

6.3 Corrección de la iluminación de la imagen

En este apartado se han utilizado tres metodologías para aplicar la corrección de la iluminación de la imagen a partir de los iluminantes detectados. Aparte de esto, cabe recalcar que se ha hecho una mejora en general sobre la corrección de la imagen tanto en la parte de un solo iluminante como en la de múltiples iluminantes. Dicha mejora consiste en el hecho de qué antes se perdía la intensidad de brillo de la imagen corregida respecto a la imagen original aparte de aplicar la corrección que sustraía el iluminante. La mejora viene dada por la Eq.7 y Eq.8.

$$Equalizer = mean(ColourConstantImage)/mean(InputImage) \quad (7)$$

$$ColourConstantImage = ColourConstantImage./Equalizer \quad (8)$$

Equalizer es el factor que ayuda a mantener el brillo de la imagen constante después de la corrección del iluminante.

Ahora se explicarán en detalle las 3 metodologías de corrección de la imagen a partir de los iluminantes detectados. En el apartado de resultados se mostrarán las 3 metodologías aplicadas sobre una misma imagen para poder hacer una comparación y ver cual funciona mejor.

6.3.1 Corrección de imagen 1

Esta metodología consiste en coger la imagen a tratar y corregirla con ambos iluminantes y finalmente aplicar Eq.7 y Eq.8 para balancear el brillo.

6.3.2 Corrección de imagen 2

Esta metodología consiste en la *gray world assumption*, es decir, coger la imagen a tratar y hacer la media de cada una de las componentes (R,G,B) de todos los píxeles y después calcular la distancia de esa media con cada iluminante. Una vez hecho esto se mira cual de las dos distancias es la más pequeña y se usa el iluminante al que corresponda para corregir la imagen y finalmente se aplican Eq.7 y Eq.8 para balancear el brillo.

6.3.2 Corrección de imagen 3

Esta metodología consiste en coger y aplicar una variante de la corrección *white patch* a la imagen a tratar cogiendo un porcentaje de los píxeles con el módulo más alto y hacer una media en cada una de las componentes (R,G,B) a partir de los píxeles de ese conjunto. Un vez hecho se calcula la distancia de la media hacia cada iluminante y se usa el iluminante más cercano a la media para corregir la imagen. Al finalizar la corrección de todos los píxeles se aplican Eq.7 y Eq.8 para balancear el brillo.

7 Experimentos y Resultados

En este apartado respecto a la detección de múltiples iluminantes se usan los datasets Gisenji et al [7] y MIRF [8] para hacer pruebas de performance del algoritmo con ambas metodologías explicadas en los apartados 6.2.1 y 6.2.2.

Para medir la performance se compara los iluminantes detectados en cada imagen con los iluminantes de la imagen de groundtruth correspondiente usando dos métricas basadas en el cálculo de angulos entre una imagen y la otra. Estas dos métricas son el *recovery angular error* definido como

$$\epsilon^{\circ}_{recovery}(e_e, e_t) = cos^{-1}\left(\frac{e_e.e_t}{\|e_e\|.\|e_t\|}\right) \quad (9)$$

donde $e_e \cdot e_t$ producto escalar entre el iluminante estimado y el iluminate del groundtruth y $\|.\|$ es la norma Euclideana. La otra métrica de performance usada es el *reproduction angular error* que se concebió como una mejora del *recovery angular error* debido a críticas de su fiabilidad que llevaron a la comunidad a mejorar la métrica anterior. El *reproduction angular error* es definido como

$$\epsilon^{\circ}_{reproduction}(e_e, e_t) = cos^{-1}\left(\frac{(e_t/e_e)}{\|e_t/e_e\|}.w\right) \quad (10)$$

donde $w = \frac{e_t/e_t}{\sqrt{3}}$ es el color de verdad de la referencia de blanco. Las tres medidas utilizadas en conjunto con la métrica para dar los resultados son la mean, la median i la trimmean. Las dos últimas medidas son especialmente buenas a la hora de dar resultados debido al hecho de que gestionan mejor los casos excepcionales.

Respecto a la corrección de los iluminantes se escogerá una imagen y se verán los resultados de aplicar una corrección u otra.

7.1 Pruebas de ejecución con dimensiones n x n

En este apartado se presentan tablas del *reproduction angular error* y del *reproduction angular error* con la mean, la median y la trimmean, usando la metodología 1 y 2 de la detección de múltiples iluminantes con las particiones fijadas en n x n. Los datos de las tablas son la mean de las tres medidas mencionadas de todos los resultados obtenidos por set de fotos con el método en cuestión.

7.1.1 Metodología 1

Tabla 2. Resultados conjunto lab recovery angular error method 1 MIRF dataset.

lab method 1 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	3	2,6	2,9
3 x 3	3,6	3,4	3,5
4 x 4	3,2	2,9	3,1
5 x 5	3,8	3,7	3,8
6 x 6	3,5	3,3	3,4

Tabla 3. Resultados conjunto real world recovery angular error method 1 MIRF dataset.

real world method 1 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	4,9	4,2	4,6
3 x 3	4,7	4,3	4,4
4 x 4	5	4,8	4,8
5 x 5	5,2	5	5
6 x 6	5,2	5	4,9

Tabla 4. Resultados conjunto lab reproduction angular error method 1 MIRF dataset.

lab method 1 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	3,1	2,7	3
3 x 3	3,8	3,6	3,7
4 x 4	3,3	3,1	3,2
5 x 5	4,1	3,9	4
6 x 6	3,7	3,5	3,6

Tabla 5. Resultados conjunto real world reproduction angular error method 1 MIRF dataset.

real world method 1 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	5,5	4,8	5,1
3 x 3	5,3	4,9	4,9
4 x 4	5,8	5,6	5,6
5 x 5	5,8	5,6	5,6
6 x 6	5,8	5,6	5,5

Tabla 6. Resultados conjunto lab recovery angular error method 1 Gisenji et al dataset.

lab method 1 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	14,4	12,8	13,5
3 x 3	13,1	12	12,2
4 x 4	12,7	11,4	11,8
5 x 5	13,1	11,8	12,2
6 x 6	12,3	10,8	11,4

Tabla 7. Resultados conjunto real world recovery angular error method 1 Gisenji et al dataset.

real world method 1 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	8,1	7,4	8
3 x 3	7,5	7,3	7,4
4 x 4	6,9	6,3	6,8
5 x 5	7	6,2	6,9
6 x 6	6,5	5,9	6,4

Tabla 8. Resultados conjunto lab reproduction angular error method 1 Gisenji et al dataset.

lab method 1 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	17,6	16,3	16,7
3 x 3	14,3	12,6	13,1
4 x 4	14,6	13,2	13,5
5 x 5	14,5	12,8	13,3
6 x 6	14,3	12,6	13,1

Tabla 9. Resultados conjunto real world reproduction angular error method 1 Gisenji et al dataset.

real world method 1 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	14,9	13,4	14,4
3 x 3	14	12,3	13,5
4 x 4	11,8	10,2	11,4
5 x 5	12,4	10,9	11,9
6 x 6	10,6	9	10,1

7.1.2 Metodología 2

Tabla 10. Resultados conjunto lab recovery angular error method 2 MIRF dataset.

lab method 2 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	2,9	2,6	2,8
3 x 3	3,2	3	3,1
4 x 4	3,3	3,1	3,2
5 x 5	3,3	3,1	3,2
6 x 6	3,5	3,3	3,4

Tabla 11. Resultados conjunto real world recovery angular error method 2 MIRF dataset.

real world method 2 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	5,1	4,6	4,8
3 x 3	5	4,7	4,8
4 x 4	5,3	5,1	5,1
5 x 5	5,1	4,8	4,8
6 x 6	5,2	4,8	4,9

Tabla 12. Resultados conjunto lab reproduction angular error method 2 MIRF dataset.

lab method 2 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	3,1	2,8	2,9
3 x 3	3,4	3,2	3,3
4 x 4	3,5	3,3	3,4
5 x 5	3,6	3,4	3,5
6 x 6	3,7	3,5	3,6

Tabla 13. Resultados conjunto real world reproduction angular error method 2 MIRF dataset.

real world method 2 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	5,4	4,7	5
3 x 3	6	5,7	5,7
4 x 4	5,8	5,7	5,6
5 x 5	5,8	5,5	5,5
6 x 6	5,9	5,6	5,6

Tabla 14. Resultados conjunto lab recovery angular error method 2 Gisenji et al dataset.

lab method 2 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	15,5	14,2	14,7
3 x 3	16,1	15	15,4
4 x 4	14,4	13,2	13,6
5 x 5	14,5	13,3	13,7
6 x 6	12,5	11	11,5

Tabla 15. Resultados conjunto real world recovery angular error method 2 Gisenji et al dataset.

real world method 2 recovery angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	8,1	7,3	8
3 x 3	7,6	7,4	7,5
4 x 4	8,1	7,4	7,9
5 x 5	6,6	6	6,5
6 x 6	6,5	5,8	6,4

Tabla 16. Resultados conjunto lab reproduction angular error method 2 Gisenji et al dataset.

lab method 2 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	18,8	17,6	17,9
3 x 3	19,8	19,1	19,1
4 x 4	16,8	15,5	15,8
5 x 5	17	15,7	16,1
6 x 6	14,9	13,1	13,7

Tabla 17. Resultados conjunto real world reproduction angular error method 2 Gisenji et al dataset.

real world method 2 reproduction angular error			
n x n	mean [◦]	median [◦]	trimmean [◦]
2 x 2	14	12,4	13,5
3 x 3	13,9	12,3	13,4
4 x 4	13,5	11,9	13
5 x 5	12,6	11	12,2
6 x 6	10,6	9	10,1

7.1.3 Comparativas de performance

Tabla 18. Comparación de performance del algoritmo propuesto por MIRF en contra de mi proyecto (2x2 + method 1) respecto al recovery angular error usando su dataset.

	median [◦]	
dataset	MIRF	My project
MIRF lab	2,6	2,6665
MIRF real world	3,4	4,2914

Tabla 19. Comparación de performance del algoritmo propuesto por Gisenji et al en contra de mi proyecto (6x6 + method 1) respecto al recovery angular error usando su dataset.

	median [◦]	
dataset	Gijsenji	My project
Gijsenij et al lab	13,4	10,8642
Gijsenij et al real world	5,6	5,9084

7.2 Pruebas de corrección de imagen

Fig. 6. Imagen antes de ser tratada.

Fig. 7. Imagen tratada con la corrección 1.

Fig. 8. Imagen tratada con la corrección 2.

Fig. 9. Imagen tratada con la corrección 3.

8 Discusión de resultados

Observando los resultados en 7.1.1 de la Tabla. 2 y la Tabla. 4, que corresponden al dataset MIRF lab se puede ver que para ambas métricas en todos los casos (n x n) hay un ángulo bastante pequeño de error, lo cual hace considerar que la detección ha sido exitosa. Aunque la detección parece funcionar bien, no parece haber una relación de mejora según se aumentan o se disminuye el número de regiones n x n. En cambio en las tablas 3 y 5 que corresponden al dataset MIRF real world el aumento de regiones parece empeorar la detección. Si ahora miramos las tablas de 6 a 9 correspondientes a los dataset Gisenji et al tanto lab como real world parecen aumentar la precisión de la detección según aumentan las particiones.

Observando los resultados en 7.1.2 de la misma forma que en el apartado 7.1.1 las tablas dan ángulos de error similares en ambas métricas y de la misma forma no parece haber una relación de mejora o empeoramiento del

ángulo de error con relación a las regiones n x n en las que se divide la imagen para los dataset de MIRF. En la metodología 2 hay el factor de qué los desplazamientos dependen de las dimensiones del tamaño de celda y las ecuaciones Eq. 5 y Eq. 6. Eso produce que puede que a veces en una cierta n x n pueda haber más desplazamientos que en otra y eso aumenta el número de detecciones de un solo iluminante.
Por último en las tablas 18 y 19 hay una comparación de rendimiento con los mejores resultados obtenidos en mi proyecto con todos los datasets respecto a los resultados obtenidos con el proyecto para el que se destinó inicialmente el dataset. Parece que mi proyecto solo es mejor sobre el dataset Gisenji et al lab, aunque en ningún caso aparte del nombrado hay una gran diferencia de rendimiento.

Respecto al apartado 7.2 las tres metodologías parecen hacer una buena corrección de imagen, pero las dos últimas dan un resultado muy parecido entre sí en comparación al resultado que da la metodología 1. La imagen usada para esta prueba pertenece al dataset MIRF lab.

9 Conclusiones

Comparando ambas metodología parece haber poca diferencia de rendimiento a nivel de detección de los iluminantes en ninguno de los sets si compararamos las tablas de cada apartado con su correspondiente en el otro. En el punto en el que sí que hay diferencia es en la carga de computo de metodologías. En la segunda, según aumenta las regiones n x n también aumenta el tiempo de ejecución respecto a la primera. Dado este dato, es mejor usar la metodología 1 a la hora de hacer la detección de múltiples iluminantes a no ser que se diponga de una gran capacidad de computo.

Respecto a la corrección de imagen debido al hecho de qué la metodología 1 hace la corrección usando los dos iluminantes para todos los píxeles y en cambio las otras dos usan la distancia euclidiana para decidir cual de los dos usar en la corrección eso causa que en las dos últimas metodologías el resultado sea parecido entre ellas, sobretodo si uno de los iluminantes es mucho más dominante sobre el otro en la imagen porque eso hace que tenga una distancia euclideana menor al otro respecto a los píxeles de la imagen original.

9 Agradecimientos

A lo largo del desarrollo de este trabajo, he tenido un gran apoyo por parte de mi tutor C. Alejandro Parraga. Hemos hecho bastantes tutorías que me han ayudado mucho a avanzar cuando estaba encallado y sin las cuales muy probablemente no habría conseguido especificar bien que tareas llevar a cabo en el desarrollo del proyecto.

También quiero agradecer el tiempo que me han dedicado a mis amigos Fermí Barlam, Adrià García Castejón, los cuales me ayudaron a debugar partes del trabajo y me ayudaron a pensar en algunos casos como hacer alguna parte del trabajo cuando estaba encallado.

Por último quiero agradecer a Arash Akbarinia por la ayuda proporcionada con una tutoría que hicimos para hablar sobre el proyecto de detección de un solo iluminante usado como proyecto base a mejorar para mi proyecto y que también me ayudara a preparar las funciones de las métricas de angular error que he utilizado para hacer pruebas de rendimiento de mi proyecto.

10 Referencias

[1] M. Ebner. (2007). Introduction. *Color Constancy* (pp. 1-8). Universität Wurzburg, Germany. John Wiley & Sons.

[2] M. Ebner. (2007). The Visual System. *Color Constancy* (pp. 9-38). Universität Wurzburg, Germany. John Wiley & Sons.

[3] M. Ebner. (2007). Color Spaces. *Color Constancy* (pp. 87-93). Universität Wurzburg, Germany. John Wiley & Sons.

[4] M. Ebner. (2007). Algorithm for Color Constancy under Uniform Illumination. *Color Constancy* (pp. 103-113). Universität Wurzburg, Germany. John Wiley & Sons.

[5] A. Akbarinia and C. A. Parraga. (August 2016). *Colour Constancy Beyond the Classical Receptive Field*. Journal of Latex class files. Vol(14), No(8).

[6] D. Cheng and others. (2016). *Two Illuminant Estimation and User Correction Preference.* IEEE Conference on Computer Vision and Pattern Recognition.

[7] Multiple light sources Dataset (2012).
http://colorconstancy.com/?page_id=21#multiplelightsourcesTIP
Last checked: 20/12/2017

[8] Multi-Illuminant Multi-Object (MIMO) (2014).
http://colorconstancy.com/?page_id=21#mimo
Last checked: 20/12/2017

[9] K-means MATLAB.
https://es.mathworks.com/help/stats/kmeans.html
last checked: 01/01/2018

[10] Gaussian functions
https://en.wikipedia.org/wiki/Gaussian_function
Last checked: 03/01/2018

[11] S. Bianco and others. (2015). *Single and Multiple Illuminant Estimation Using Convolutional Neural Networks.*

https://arxiv.org/pdf/1508.00998v2.pdf
Last checked: 03/01/2018

[12] Reproduction Angular Error: An Improved Performance Metric for Illuminant Estimation
http://www.bmva.org/bmvc/2014/files/paper047.pdf
Last checked: 02/02/2018

[13] Multi-Illuminant Estimation with Conditional Random Fields
http://refbase.cvc.uab.es/files/BRW2014.pdf
Last checked: 07/02/2018

[14] Color Constancy for Multiple Light
https://ivi.fnwi.uva.nl/isis/publications/2012/GijsenijTIP2012/GijsenijTIP2012.pdf
Last checked: 07/02/2018

11 Apéndice

Tabla 20. Recovery angular error de cada iluminante con los parámetros que dieron mejor resultado en el recovery angular error global (2 x 2 + method 1) del dataset MIRF lab.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
1,2	2,2	1,3	2,6

Tabla 21. Recovery angular error de cada iluminante con los parámetros que dieron mejor resultado en el recovery angular error global (2 x 2 + method 1) del dataset MIRF real world.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
2	2,8	3,1	3,3

Tabla 22. Reproduction angular error de cada iluminante con los parámetros que dieron mejor resultado en el reproduction angular error global (2 x 2 + method 1) del dataset MIRF lab.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
1,3	2,3	1,4	2,8

Tabla 23. Reproduction angular error de cada iluminante con los parámetros que dieron mejor resultado en el reproduction angular error global (2 x 2 + method 1) del dataset MIRF real world.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
2,3	2,5	3,7	3,3

Tabla 24. Recovery angular error de cada iluminante con los parámetros que dieron mejor resultado en el recovery angular error global (6 x 6 + method 1) del dataset Gisenji et al lab.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
5,6	15	8,6	16

Tabla 25. Recovery angular error de cada iluminante con los parámetros que dieron mejor resultado en el recovery angular error global (6 x 6 + method 1) del dataset Gisenji et al real world.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
8,5	4,1	11	5,2

Tabla 25. Reproduction angular error de cada iluminante con los parámetros que dieron mejor resultado en el reproduction angular error global (6 x 6 + method 1) del dataset Gisenji et al lab.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
9,4	20,4	11,5	21,3

Tabla 25. Reproduction angular error de cada iluminante con los parámetros que dieron mejor resultado en e lreproduction angular error global (6 x 6 + method 1) del dataset Gisenji et al real world.

median [◦]		trimmean [◦]	
illuminant 1	**illuminant 2**	**illuminant 1**	**illuminant 2**
8,5	4,3	10,9	5