Universidad de Oviedo

Memoria del Trabajo Fin de Máster realizado por

ANDRÉS GARCÍA CORBATO

para la obtención del título de

Máster en Ingeniería de Automatización e Informática Industrial

APLICACIÓN DE COLORIMETRÍA PARA CORRECCIÓN DE IMÁGENES

JUNIO 2015

Aplicación de colorimetría para corrección de imágenes

Uso de un sensor RGB para la predicción del color inducido por la fuente de iluminación

Andrés García Corbato

Índice

TRABAJO FIN DE MÁSTER

Máster en Ingeniería de Automatización e Informática Industrial

“Aplicación de Colorimetría para corrección de imágenes”

1. Introducción

1.1. Motivación

Los sistemas de visión artificial son de vital importancia para la industria moderna debido al continuo avance acontecido en los algoritmos de procesamiento de imágenes y, consecuentemente, de la cantidad de información que puede ser obtenida.

Para que estos sistemas ejecuten su labor de forma óptima y se facilite el procesado posterior, la calidad de la imagen capturada tiene que ser lo más alta posible según los requerimientos establecidos en cada proceso.

Cuando las imágenes son tomadas en interiores o en espacios confinados, las condiciones que repercuten sobre la toma de las mismas pueden ser controladas de una forma relativamente sencilla, al contrario de lo que ocurre cuando se obtienen imágenes en ambientes exteriores, donde se carece de tal control. En cualquiera de las dos situaciones, la aplicación de métodos de corrección por software resulta de gran utilidad para el ajuste de imágenes, siendo especialmente importante para el segundo caso.

Uno de los problemas que se pueden presentar al capturar imágenes con una cámara es la presencia de un tono de color predominante (“*color casted images*”) producido por la fuente de iluminación del conjunto. Este problema es bastante grave, ya que, al contrario de cómo funciona el sistema visual humano, capaz de contrarrestar la predominante de color, las cámaras carecen de tal ajuste. La solución a este problema es el filtrado a través de algoritmos de Balance de Blancos.

Tales algoritmos se basan en su gran mayoría en métodos numérico-estadísticos que toman como parámetros los valores de los píxeles capturados, pudiendo introducir aberraciones en los colores originales debido al desconocimiento de las características de la fuente de iluminación del conjunto. La existencia de dispositivos externos al sistema de captura que se encarguen de medir la luz incidente se reserva en muchos casos a fotografía artística o profesional, pero normalmente no se aplica a ambientes industriales.

Por otro lado los dispositivos de medición de las características de la luz incidente suelen tener un precio alto y carecer de flexibilidad a la hora de interactuar con los sistemas de captura (cámaras) u otros procesadores.

1.2. Objetivo del presente Trabajo

En publicaciones como [13] se identifica que sólo en el mercado de cámaras profesionales de la mayor calidad se dispone de sensores adicionales que permitan la caracterización del espectro lumínico de la fuente de iluminación. Normalmente, con el objetivo de reducción de costes, este segundo sensor es omitido; apareciendo en estos casos algoritmos de balance de blancos basados únicamente en información de la imagen capturada.

De acuerdo a [14], un algoritmo de balance de blancos genérico consiste en dos pasos esenciales, el primero sería la estimación de la temperatura de color de la fuente de iluminación, siendo el segundo el ajuste del contenido de la imagen para compensar tal efecto de la iluminación.

Es por ello que, dada la carencia de sistemas de ajuste de color en imágenes basadas sobre mediciones externas al dispositivo de captura (cámara), el objetivo del presente trabajo será:

"Adaptar un sistema de medición de las características de la luz, especialmente para aquellas imágenes con una fuerte presencia de inducción de color, basado en tecnología de bajo coste. También se hará una propuesta de adaptación de un algoritmo de Balance de Blancos que permita la actuación conjunta con el antedicho dispositivo y optimice la corrección de las imágenes capturadas. Como forma de comprobación de la adecuación del desarrollo tecnológico propuesto en el presente Trabajo, se construirá un prototipo".

1.3. Desarrollo del trabajo

El desarrollo del presente Trabajo Fin de Máster se establecerá de la siguiente manera:

- En primer lugar se revisará el estado del arte de los diferentes dispositivos que permiten realizar mediciones de la luz incidente, de tal forma que se consiga localizar un dispositivo que se adecúe a las características descritas en el anterior apartado de "Motivación".
- Como sustento fundamental del presente estudio, con la información que ha sido reunida en anteriores apartados se elaborará un prototipo que permita la revisión conjunta tanto de la medición de las características de la luz como del algoritmo seleccionado.
- Finalmente se presentarán las conclusiones y resultados obtenidos acorde a los test realizados sobre el prototipo.

2. Conceptos básicos

En este apartado se detallarán aquellos conceptos teóricos que conviene que sean introducidos para mejorar la comprensión del desarrollo del presente trabajo.

2.1. Espectro de color y el ser humano

Las fuentes luminosas emiten su energía en diferentes longitudes de onda, definiendo cada longitud de onda visible un color diferente[15]. El ser humano tan sólo es capaz de visualizar un subconjunto de las longitudes de onda existentes, las que van desde los 380 nanómetros que se corresponden con el color violeta, hasta los 780 nanómetros que son las del color rojo. Ésta porción visible es la denominada espectro visible.

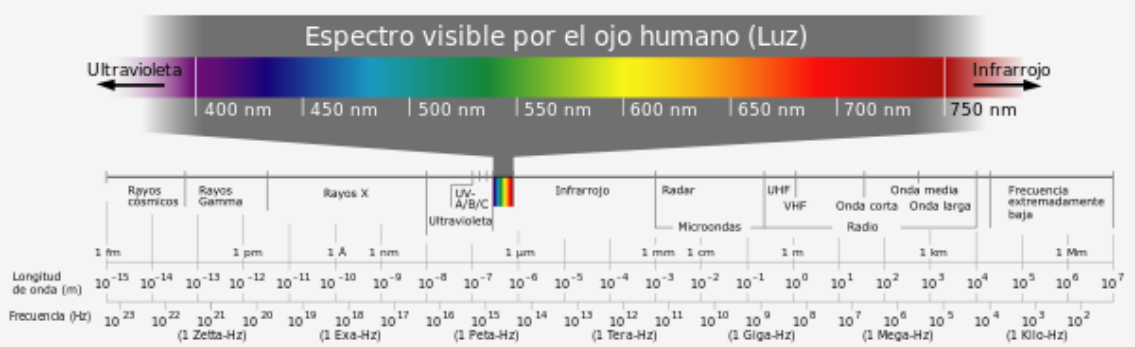


Figura 1 – *Espectro elecromagnetíco con el espectro visible en detalle*

2.2. Formación del color

La formación del color en los objetos depende de dos cosas fundamentales, la naturaleza de la luz y la composición del objeto que marcarán la absorción y la reflexión de la luz[16].

Un cuerpo opaco absorbe gran parte de la luz que lo ilumina y refleja una parte más o menos pequeña. Todos los cuerpos están constituidos por sustancias que absorben y reflejan las ondas electromagnéticas en diferentes rangos. Un cuerpo se ve blanco porque recibe todos los colores básicos del espectro y los devuelve reflejados, generándose así la mezcla de ellos y formándose el blanco. Los colores absorbidos desaparecen en el interior del objeto, mientras que los reflejados llegan al ojo humano o al dispositivo de captura utilizado. Por el contrario un cuerpo se ve negro porque absorbe todas las radiaciones electromagnéticas que indicen sobre él, no reflejando ninguno.

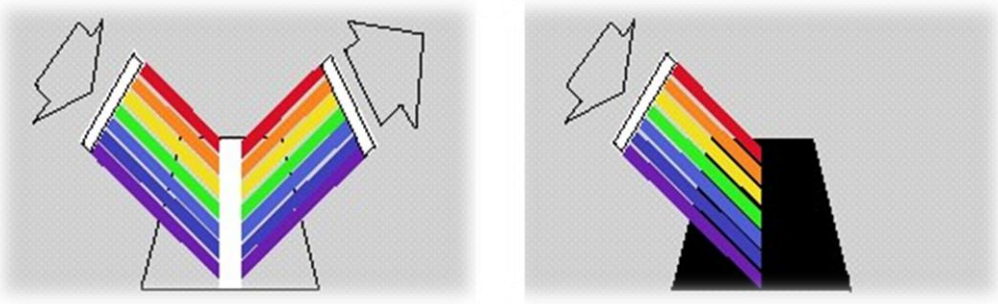


Figura 2 – *Formación del color (blanco y negro)*

El resto de colores (no blanco y negro) se forma de igual forma, es decir, son el resultado de la mezcla de las radiaciones electromagnéticas que los propios objetos no absorben y son reflejadas hacia el dispositivo de captura. Por ejemplo, el rojo se forma porque sólo las longitudes de onda en torno al rojo son reflejadas por el objeto.

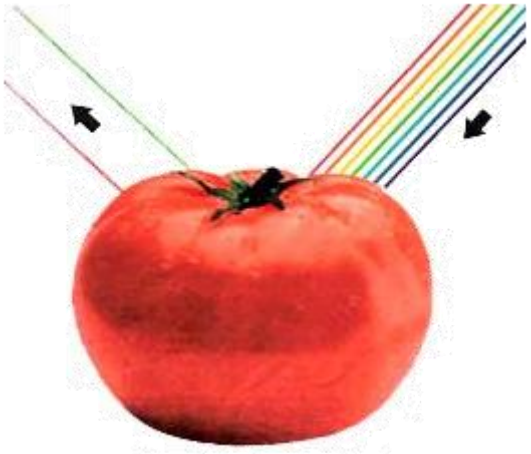


Figura 3 – *Formación de los colores rojo y verde sobre un objeto*

2.3. Balance de Blancos

El Balance de Blancos ("White Balance") es un método de ajuste de imágenes que consigue corregir las dominantes de color presentes en imágenes obteniendo por tanto, unas imágenes más cercanas a los colores reales independientemente del tipo de fuente de luz que ilumina la escena.

La siguiente figura muestra dos imágenes de la misma escena en las cuales puede observarse la predominancia de tono azul existente en una de ellas con respecto a la fotografía tomada con los valores adecuados para el balance de blancos.

Figura 4 – *Iglesia con un claro tono predominante azul (izquierda) y la misma escena tomada con un correcto balance de blancos (derecha)*

Al contrario de lo que ocurre con el sistema visual humano, en el cual los tonos blancos se perciben como blancos incluso cuando tienen algún tipo de componente de color inducida, una cámara que no tenga el balance de blancos ajustado percibirá los objetos con un color que dependerá de la fuente de luz que los esté iluminando [1]. Por ejemplo, iluminando con luz fluorescente se obtendrán matices verdes, mientras que si la fuente de luz es incandescente esa componente será anaranjada.

En conclusión, el balance de blancos se basa en hacer que los objetos blancos se perciban blancos en la imagen final con independencia del color inducido por la fuente de iluminación. Al contrario de lo que sucede en fotografía artística, donde este ajuste puede ser modificado para obtener efectos diferentes, en el ámbito industrial y en inspección en particular el principal efecto buscado es la captación de los colores verdaderos, es decir, tal y como son en la realidad.

2.3.1. Importancia del balance de blancos en exteriores

En exteriores, durante el día, la fuente de iluminación principal es el sol (a no ser que exista otro tipo de fuente como focos, antorchas, etc.). En este caso la importancia del Balance de Blancos reside en las variaciones que el sol como fuente puede sufrir debido a la hora del día, la fase del año o efectos climatológicos o efectos de la dispersión de la luz en la atmósfera entre otros.

La siguiente tabla muestra las variaciones (de forma aproximada) que sufre la luz del sol sobre exteriores [2]. El color se muestra en forma de grados Kelvin que serán introducidos en el siguiente punto de este documento.

Tabla 1 – *Variación aproximado del color de la luz solar*

Luz del día (Combina la luz del Sol y la proveniente de la atmósfera) medida en K	
Luz del Sol: salida y puesta del sol	**2000K**
Luz del Sol: una hora después del amanecer	**3500K**
Luz del Sol: temprano por la mañana o tarde de tarde	**4300K**
Luz del Sol media en verano al mediodía en las latitudes medias	**5400K**
Luz del Sol directa a mitad de verano	**5800K**
Cielo nublado	**6000K**
Luz del Sol media en verano (con cielo azul)	**6500K**
Sombra con luz de verano	**7100K**
Sombra media en verano	**8000K**
Luz del cielo (atmósfera) en verano (varía)	**9500-30000K**

2.4. Temperatura de Color (Correlated Color Temperature)

Un concepto de gran relevancia en relación a la colorimetría es la medida de la "Temperatura de Color en su escala en grados Kelvin". Este concepto está relacionado con la fuente de luz y se define como la comparación de la apariencia en color de la fuente con la de la luz que emitiría un Cuerpo Negro perfecto calentado a una temperatura determinada. En el diagrama de cromaticidad del CIE1931, que representa un espacio de color determinado presentado en la siguiente figura, se puede encontrar de forma directa la representación del lugar "Planckiano" o Cuerpo Negro en función de las coordenadas "x" e "y" características de ese espacio (este concepto será introducido en un apartado posterior) [3].

Este parámetro presenta una ventaja fundamental y es que, a través de él se puede establecer una relación directa con un color, definido a través de sus tres componentes trabajando en un espacio de color determinado (XYZ CIE1931, RGB, etc.) [4], siendo mucho más fácil de caracterizar por tanto el color objetivo. En el caso de algunas plataformas software, como Photoshop, los balances de blancos pueden ser establecidos en base a este único parámetro.

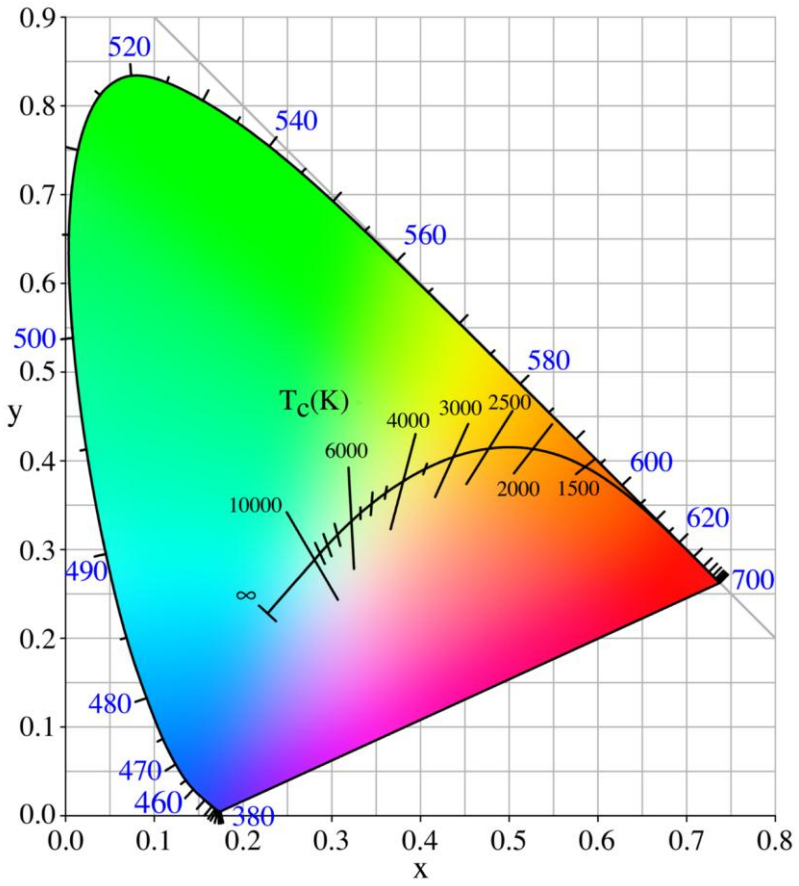


Figura 5 - *Diagrama de cromaticidad según CIE, con el lugar "Planckiano" (Cuerpo Negro) definiendo la temperatura de color*

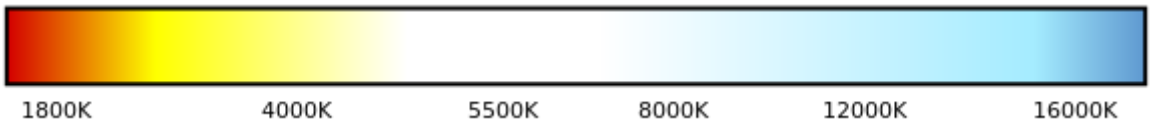


Figura 6 – *Representación aproximada de los colores a través de la temperatura de color (escala de color en grados Kelvin)*

2.5. Calibrador de color Macbeth

Según se establece en [10], el calibrador de color o muestrario Macbeth es un estándar industrial que provee una comparación no subjetiva a través de una muestra de 24 cuadrados de color científicamente preparados y calibrados, manteniendo así la consistencia de color sobre diferentes

opciones de iluminación. También se respalda esta teoría en [12], asegurando que gracias a la existencia de un "Color Checker" en las imágenes de muestra se pueden detectar los valores verdaderos para realizar correcciones en las imágenes.

Es por ello que para esta plantilla de colores se pueden identificar dos usos primarios, por un lado el **ajuste de imágenes** a través de la presencia de la misma dentro de las imágenes capturadas (patrón calibrado dentro de la imagen que se captura, que permite determinar las desviaciones en los colores reales producidas por la fuente de iluminación) que más tarde van a ser corregidas, pudiendo extenderse esta funcionalidad a la **calibración** del método de ajuste, mientras que por otro lado puede utilizarse para realizar una evaluación objetiva de los métodos de balance de blancos y ajuste de color que se apliquen en el proceso de captura de las mismas.

Ajuste de imágenes

En el caso del ajuste de las imágenes, uno de los exponentes de esta teoría puede encontrarse en [9], donde se particulariza que en las aplicaciones de visión por computador es muy importante que los colores que son capturados permanezcan constantes con respecto a la escena real, incluso con cambios de iluminación o de cámaras.

Extendiendo la funcionalidad [11], gracias a la correlación entre el ajuste obtenido a través del "color checker" y el valor medido que caracterice la luz incidente, se podrá calibrar el método de ajuste así como el sensor utilizado para la medición evaluativa de la luz.

Evaluación objetiva de ajustes

En el artículo [10], los algoritmos de balance de blancos son evaluados de dos formas, objetiva y subjetiva. La forma subjetiva se basa en métodos de evaluación visual y percepción humana de los colores observados en la escena, mientras que para la verificación objetiva se utilizan imágenes obtenidas con la plantilla de colores de Macbeth o "color checker" insertado, siendo las mismas tomadas bajo diferentes condiciones de iluminación.

Figura 7 – *Escena tomada bajo diferentes condiciones de iluminación (Iluminador A, TL84, U30 y D50) con Macbeth color checker insertado*

Parece claro que el método más interesante es el método objetivo, ya que su reproducibilidad no será determinada por la sensibilidad visual inherente a cada persona humana, sino que su utilidad se basa en la inserción de un testigo calibrado. La evaluación realizada en tal publicación fue la siguiente: como la reflectancia de la plantilla de Macbeth está tasada y es conocida, los valores RGB en el espacio de color Adobe son usados como datos de referencia para determinar la eficiencia de los algoritmos de balance de blancos. En este caso el criterio de selección se basa en la proximidad de los valores RGB a las referencias que vienen dadas en la tabla de calibración.

Estimated RGB values of Macbeth chart under different Illuminations after AWB algorithms

Gray patches	RGB (Babel)	Illuminant A		U30		TL84		D50		D65		D75	
		4[1]	7[1]	1[1]	7[1]	5.2[1]	6[1]	2[1]	7[1]	5.1[1]	5.2[1]	2[1]	5.1[1]
R	245	200	221	195	194	222	216	208	208	217	217	213	211
G	245	208	234	204	206	220	222	205	219	217	217	212	212
B	242	218	232	212	196	213	216	201	216	217	216	207	215
R	200	174	192	168	167	193	187	179	178	189	188	183	181
G	201	181	204	175	177	192	194	176	189	187	187	184	184
B	201	186	198	177	164	186	188	174	186	190	190	181	188
R	160	145	160	136	135	159	155	145	144	156	156	150	148
G	161	150	169	143	144	158	160	143	153	154	154	150	150
B	162	151	161	137	127	150	155	138	148	154	154	146	151
R	120	105	116	96	96	114	111	101	101	111	110	104	104
G	120	105	118	99	100	112	113	98	105	108	108	105	104
B	121	103	109	91	84	105	107	94	101	107	107	100	104
R	84	58	64	48	47	60	59	52	51	56	56	54	53
G	85	60	68	49	50	60	61	50	54	55	55	54	54
B	86	61	65	49	45	59	59	52	56	59	58	55	57
R	52	26	28	24	24	27	27	26	26	27	27	26	26
G	53	25	28	25	25	27	27	25	27	26	26	26	26
B	54	25	27	26	26	28	28	28	30	28	28	27	28

[1] 1 - GWT, 2 - RM, 3 - GWR, 4 - SDWGW, 5.1 - $SDLWGW_1$, 5.2 - $SDLWGW_2$, 6 - SDL, 7 - WPYCC

Figura 8- *Resultados de la estimación de los colores en la plantilla de calibración bajo diferentes condiciones de iluminación.*

La plantilla consiste [11] en 24 muestras de color formuladas para emular los colores naturales como los de la piel, el follaje y el cielo, además de seis tonos de una escala de gris. Las componentes en sRGB en formato de 8-bits y L*a*b D50 son proporcionadas, mientras que la conversión a otros espacios de color no está disponible. En el artículo citado se pueden encontrar tabulados los valores para los espacios de color adicionales: Adobe RGB, Apple RGB y ProPhoto, que puede resultar de gran utilidad al utilizar este tipo de calibradores en otro tipo de aplicaciones o cuando se trabaje en tales espacios de color.

Aunque el calibrador presentado puede resultar de gran utilidad a la hora de la evaluación de la efectividad de los algoritmos de calibración de colores, el objetivo final no es encontrar el mejor algoritmo

de balance de blancos, sino que será el conseguir una calibración aceptable del sensor; que unido a que **esta plantilla**, debido a su calibración, **tiene un precio relativamente alto** (aproximadamente 200€), no será utilizado en el presente trabajo, pero **será recomendado para posteriores avances en la misma línea**.

No obstante, a pesar de que el método objetivo es el más adecuado, al no disponer del "color checker" calibrado (una impresión en color será utilizada como indicativa) para la elaboración del presente trabajo, la evaluación subjetiva será considerada como válida para la consecución del objetivo: "apreciar la capacidad de mejora del algoritmo que se quiera testear".

2.6. Espacios de Color

Otro concepto que hay que introducir antes de adentrarse más a fondo en la corrección de imágenes es el de los espacios o modelos de color. Este modelo de color [5] es un modelo matemático abstracto que describe la forma en la que los colores pueden representarse como tuplas de números, normalmente como tres o cuatro valores o componentes de color; es decir, el objetivo de un modelo de color [6] es facilitar la especificación de los colores de una forma normalizada y aceptada genéricamente; es la especificación de un sistema de coordenadas tridimensionales y de un subespacio de este sistema en el que cada color queda representado por un único punto.

2.6.1. CIE XYZ y CIE xyY

La CIE [5] (Comisión Internacional de Iluminación, Comission Internationale de l'Éclairage) es la autoridad internacional en cuestiones de luz, iluminación, color y espacios de color. Esta comisión estableció en los años 30 una serie de normas para los diferentes espacios de color que representan el espectro visible y gracias a estas normas se pueden hacer comparaciones entre los diversos espacios de color de los visores y dispositivos.

El sistema de color XYZ o estándar se desarrolló para ser usado como referencia para definir los colores que percibe el ojo humano y otros espacios de color. Este espacio se basa en tres primarios imaginarios con caracterización espectral (X, Y y Z), que son los que representan el color (ondas electromagnéticas) combinándose para formar todos los colores visibles por el "observador estándar".

En el modelo descrito, Y significa luminosidad, Z es aproximadamente igual al estímulo de azul (conos S) y X es una mezcla con la curva de sensibilidad del rojo (conos M y L). Como el ojo humano tiene tres tipos de células receptoras de color, que se estimulan ante distintos rangos de longitud de onda, una carta completa de todos los colores visibles es realmente una figura tridimensional, de tal forma que el concepto de color puede ser dividido en brillo y cromaticidad.

En este espacio de color el parámetro Y es una medida del brillo o luminosidad de un color mientras que la cromaticidad de dicho color se determina a través de dos parámetros derivados x e y, que vienen de

la normalización de los valores primarios X, Y y Z. Tales valores primarios tienen las siguientes propiedades [7]:

- Siempre producen un valor triestímulo positivo.
- Se puede representar cualquier color en los términos de estas primarias.
- Están relacionadas con la sensibilidad el ojo humano por el uso de funciones de igualación ("Color Matching Functions") que se corresponde con el observador estándar según el CIE1931.

Si la distribución de la respuesta espectral ("Spectral Power Distribution" SPD) de un objeto de color se pondera por las siguientes curvas, que son las funciones de igualación antedichas, se pueden calcular las coordenadas de cromaticidad XYZ:

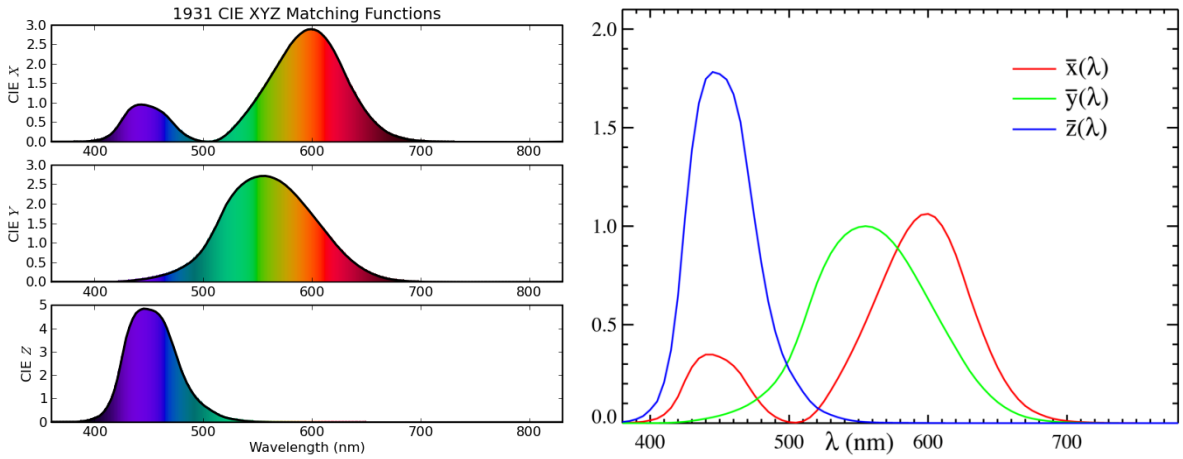


Figura 9 – *Funciones de igualación según el observador estándar en CIE1931*

Estas curvas o funciones de igualación del observador estándar proveen de una forma de convertir la radiación espectral L(λ) a un espacio de color tri-cromátrico, establecido a través de la siguiente ecuación.

$$X = \int L(\lambda)\,\bar{x}(\lambda)\,d\lambda,$$
$$Y = \int L(\lambda)\,\bar{y}(\lambda)\,d\lambda,$$
$$Z = \int L(\lambda)\,\bar{z}(\lambda)\,d\lambda.$$

Ecuación 1 – *Conversión al espacio tri-cromático*

Para obtener los valores derivados normalizados "x,y y z", la relación establecida es la mostrada en la siguiente ecuación.

$$x = \frac{X}{X+Y+Z}$$
$$y = \frac{Y}{X+Y+Z}$$
$$z = \frac{Z}{X+Y+Z} = 1 - x - y$$

Ecuación 2 – *Obtención de los parámetros normalizados derivados*

2.6.2. Espacio RGB

Debido a características inherentes del ojo humano y la teoría tricrómica, todos los colores que pueden ser reconocidos en una imagen son combinaciones de aquellos llamados colores primarios Rojo (R), Verde (G) y Azul (A) [6].

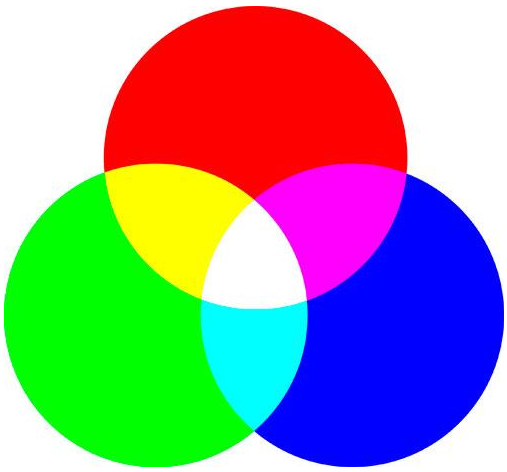


Figura 10 - *Representación de los colores primarios*

El modelo RGB es uno de los más recursivamente utilizados. En él cada color aparece en sus componentes espectrales primarias: rojo, verde y azul, y está basado en el sistema de coordenadas cartesianas. El subespacio de color es el tetraedro mostrado en la siguiente figura:

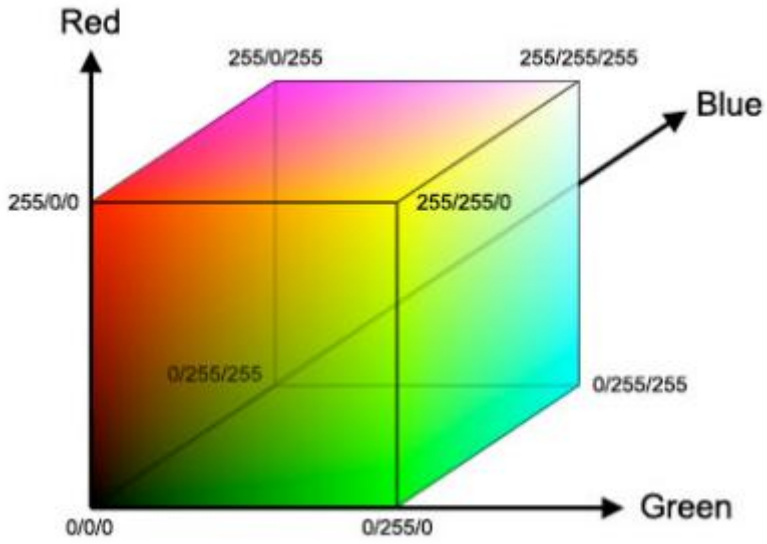


Figura 11 – *Tetraedro del sistema RGB basado en coordenadas cartesianas (resolución 8 bits)*

En el anterior tetraedro los valores de RGB están en tres vértices mientras que cian, magenta y amarillo se sitúan en otros tres. El reparto de vértices se completa de forma que el negro corresponde al origen y el blanco al extremo más alejado del origen.

- Rojo: [1,0,0]
- Verde: [0,1,0]
- Azul: [0,0,1]
- Cian: [0,1,1]
- Magenta: [1,0,1]
- Amarillo: [1,1,0]
- Blanco: [1,1,1]
- Negro: [0,0,0]

La escala de grises se extiende desde el negro al blanco sobre la diagonal formada al unir ambos vértices, siendo los colores puntos dentro del tetraedro definidos por los vectores desde el origen. Por norma general, se asume que todos los vectores han sido normalizados, de modo que el tetraedro es unitario estando los valores de RGB en el rango [0,1].

Las imágenes constituidas en el espacio de color RGB son el resultado de la suma de tres planos de imagen independientes, constituido cada uno de ellos por el nivel de cada uno de los colores primarios RGB. La mayoría de cámaras fotográficas utilizan este espacio de color para capturar las imágenes, descomponiendo el color en los niveles por píxel de cada uno de los colores primarios, bien a través de filtros Bayer (las que poseen un único sensor CCD) o bien existen cámaras que poseen tres sensores CCD, cada uno encargado de recoger los niveles por píxel de cada color primario RGB.

Figura 12 - *Muestra de la descomposición en RGB de una imagen*

3. Medición de las características de la luz: prototipo y calibración

Uno de los factores más importantes de cara realizar un algoritmo de balance de blancos sobre una imagen capturada, se basa como se ha comentado con anterioridad en la medición de las características de la fuente de iluminación del conjunto como factor primordial.

Con el objetivo de conseguir desarrollar un sistema con las capacidades de medición requeridas, se propondrá y diseñará un prototipo que será calibrado y testeado a través de un desarrollo experimental definido.

3.1. Prototipo

El prototipo resultará fundamental para realizar las comprobaciones de la tecnología que va a ser utilizada para medir las características de la fuente de iluminación. La funcionalidad principal del dispositivo que va a ser diseñado en el presente trabajo será para realizar mediciones de la luz con colores inducidos fuertes, aunque también funcionará para casos con menor incidencia, tanto en interiores como en exteriores.

Como se había mencionado, el prototipo debe poseer ciertas características debido al trabajo de captura imágenes en diferentes ambientes para los que está prevista la utilización del sistema descrito a lo largo del Trabajo. Tales características se corresponden con:

- Disponer de una fuente de luz que permita realizar variaciones sobre el color dominante inducido.
- Disponer de un sensor cuya medida sirva como input para el algoritmo.
- Disponer de un módulo de procesamiento que sirva de vínculo entre la captura de imágenes y la señal de realimentación.

La siguiente imagen muestra la estructura que dispondrá a grandes rasgos el sistema en su fase de prototipo.

Figura 13 –*Arquitectura principal del prototipo*

En tal estructura se pueden diferenciar las siguientes partes:

Fuente de iluminación

- Una lámpara junto con unos filtros como la fuente de luz que ilumine el sistema.

Sistema de captura de imágenes

- Una cámara que realizará la función de captura de imágenes de la zona objetivo.

Sistema sensorial

- Un sensor RGB situado en la zona objetivo que provea al conjunto de una señal que caracterice la iluminación recibida en la citada zona.

Sistema de procesamiento

- Un microcontrolador que haga la función de pre-procesamiento y adecuación de la señal para que pueda ser procesada en un ordenador.
- El propio ordenador como unidad de procesamiento tanto de la imagen capturada por la cámara como la información recibida del pre-procesamiento de la señal recibida del sensor RGB.

A continuación se describirá en mayor grado de detalle cada uno de los componentes del sistema.

3.1.1. Fuente de iluminación

La fuente de iluminación es el un elemento primordial, ya que es imprescindible para la captura de la imagen. Es la responsable de proveer de la iluminación necesaria para que se pueda observar el objeto que se encuentre situado en la zona objetivo. Como existen gran cantidad de tipos de fuentes, es preciso realizar a priori un estudio detallado de las características que deben cumplir.

De acuerdo con [22] se puede obtener una orientación general de las características que se buscan para aplicaciones de visión por computador en color, a pesar de que la información contenida en esta fuente de información es para aplicaciones de cine y televisión en su gran mayoría.

En el proceso de selección de la fuente de iluminación, destacan tres premisas fundamentales para la evaluación, que son:

- Estabilidad de color: habilidad de la lámpara/fuente para mantener la cromaticidad estable con el paso del tiempo, es decir, la capacidad de la fuente para el mantenimiento del color que induce con el paso del tiempo.
- Consistencia de color: variación de la distribución de la potencia espectral entre varias fuentes idénticas.
- Mantenimiento de lumen: depreciación en el flujo radiante. Cambios en la salida espectral, es decir, conservación de la intensidad lumínica con el paso del tiempo.

A pesar de que las premisas mostradas con anterioridad son las fundamentales a la hora de seleccionar una fuente de iluminación, son premisas específicas para situaciones donde tal fuente (artificial) va a ser utilizada durante largos períodos de tiempo así como que en combinación con otras. En el caso del prototipo que se está tratando en este documento, la importancia de estas premisas se ve atenuada, ya que sólo se va a utilizar una única fuente y los experimentos serán realizados en periodos de tiempo cortos.

No obstante, existen otros estudios que se adecuan más a las necesidades específicas del Trabajo en cuanto a fuentes de iluminación se refiere, derivadas del análisis espectral tradicional, del cual se desprende:

- Una fuente de luz de calidad presenta un espectro suave, sin picos marcados.
- Se establece el Índice de Reproducción Cromática (CRI) como índice de referencia para la reproducción de colores de una fuente de iluminación.

Una vez entendido que el color es el resultado de la reflexión de ciertas longitudes de onda sobre el objeto sujeto a análisis, dependiendo directamente de la composición del objeto, cabe destacar la segunda parte implicada que es la respuesta espectral de las fuentes de iluminación.

De sobra es conocido que no todas las fuentes de luz tienen un comportamiento espectral igual, presentando en algunos casos un espectro suave que denota una fuente de luz de calidad, mientras que otras sufren variaciones abruptas que se traducen en picos sobre el espectro y discontinuidad en algunos casos.

El problema aparece cuando una determinada fuente carece de ciertas longitudes de onda causando que los colores percibidos no sean los que realmente tendrían bajo una iluminación idónea. Esto se debe a que, como ha sido mencionado con antelación, para que el color se forme en el sistema de captura, deben existir todas aquellas longitudes de onda que no son absorbidas por el objeto, y en el caso de que falte alguna, no se producirá el mismo efecto.

A continuación se exponen algunas respuestas espectrales de diferentes tipos de luz para comprobar las diferencias existentes. En concreto, se corresponden con las respuestas de la luz solar, fuentes fluorescentes, fuentes halógenas, fuentes incandescentes y luces LED. Tales imágenes han sido tomadas de [17] y están representadas en una escala normalizada.

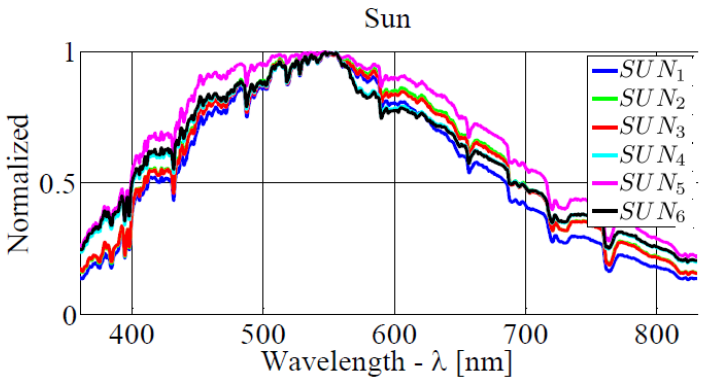


Figura 14 - *Respuesta espectral del sol al mediodía en 6 días diferentes*

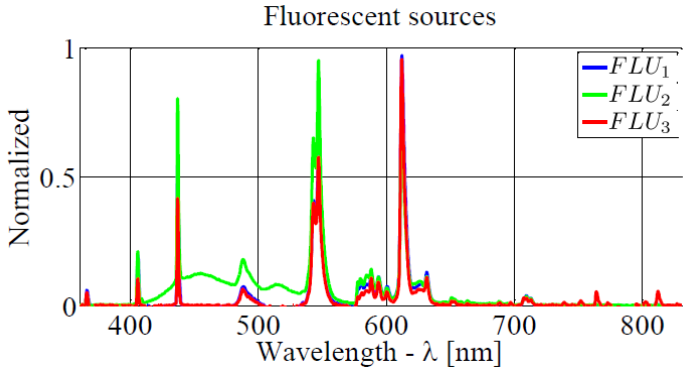


Figura 15 – *Respuesta espectral de tres fuentes fluorescentes*

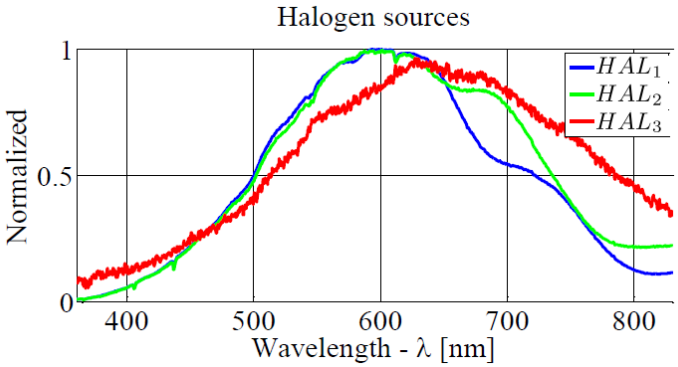


Figura 16 – *Respuesta espectral de 3 fuentes halógenas*

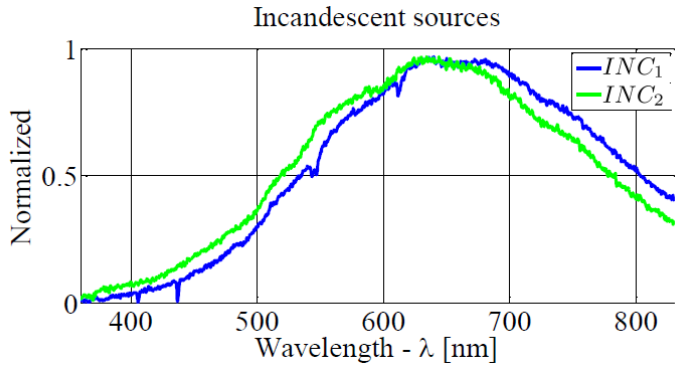


Figura 17 - *Respuesta espectral de dos fuentes incandescentes*

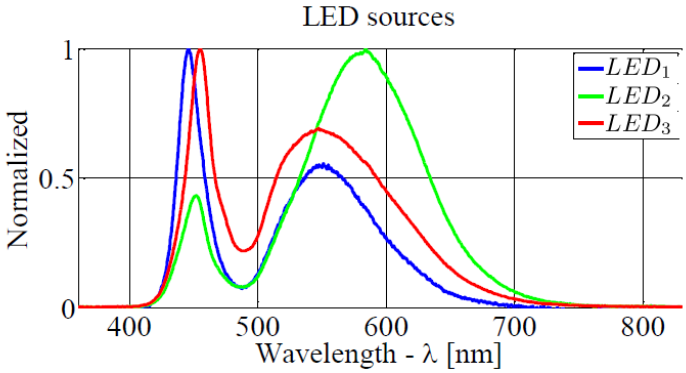


Figura 18 - *Respuesta espectral de tres fuentes LED*

Como se mencionó con anterioridad, la respuesta espectral no es el único parámetro al que atenerse a la hora de seleccionar una fuente de iluminación. El Índice de Reproducción Cromática (CRI) es una medida de la capacidad que una fuente luminosa tiene para reproducir fielmente los colores de varios

objetos en comparación con una fuente de luz natural o ideal [23] que resultará de gran utilidad para seleccionar de una forma simple la fuente de luz más adecuada.

De acuerdo con [26], la distribución espectral representa la distribución de la energía radiada en longitudes de onda diferentes en la parte visible del espectro luminoso. Esta distribución puede ser definida para cada tipo de fuente de luz, es decir, este índice está relacionado con el modo en que aparecen los objetos bajo una fuente de luz dada.

Un CRI bajo indica que los objetos pueden parecer poco naturales bajo la fuente, mientras que una luz con un índice CRI alto permitirá que los colores de un objeto parezcan más naturales. El valor máximo del CRI para una fuente es de 100, significando que toda la gama de colores se reproducen fielmente, corresponde a la luz del día [24]. Las fuentes de luz que proporcionan un CRI de más de 80 se consideran excelentes para el reconocimiento del color. A continuación se muestra una figura tomada de [25], donde de una forma muy simplificada se muestra el Índice de Reproducción Cromática de diferentes fuentes.

Figura 19 - *CRI de diferentes fuentes*

A la vista tanto de la respuesta espectral de las fuentes tanto como su Índice de Reproducción Cromática, está claro que para obtener unos resultados lo más correctos posibles se optará por utilizar una fuente **halógena**, que además de disponer de un espectro continuo y un valor próximo a 100 en CRI, es a priori una de las mejores fuentes para este objetivo.

No obstante cabe destacar que, a pesar de resultar ventajosa en los aspectos antedichos, las bombillas halógenas presentan por norma general consumos mucho mayores a los que presentan por ejemplo las bombillas LED, además de que inducen un color en torno a los 2800-3400 K (tono cálido) a las imágenes iluminadas.

3.2. Sistema Sensorial

Con el objetivo de realizar tal medición y tomando como referencia las publicaciones [17][18] se puede determinar una forma para la consecución de tal objetivo.

En ambos casos, la medición de la temperatura de color emitida por la fuente de iluminación del conjunto se realiza a través de sensores RGB, más concretamente en el modelo TCS3414CS del fabricante TAOS[19].

Las características principales de este sensor se recogen en el siguiente párrafo:

"El sensor digital de color TC3414CS está diseñado para definir de forma precisa la cromaticidad y la intensidad de la luz ambiente y proveer una salida digital con 16bits de resolución. El dispositivo incluye un array de 8x2 fotodiodos filtrados, conversores analógico-digital y funciones de control en un único circuito integrado CMOS. De los 16 fotodiodos, 4 tienen filtros rojos, 4 filtros verdes, 4 filtros azules y 4 no tienen filtro (claro)."

Figura 20 – *TAOS TCS3414CS*

El dispositivo utilizado en las antedichas publicaciones no es fácil de encontrar a la venta, por lo que se buscó una alternativa como aspecto clave para la consecución del presente trabajo. Como alternativa se localizó un sensor de funcionamiento similar, que es el TCS34725. La descripción general de tal dispositivo es la siguiente [20]:

"El dispositivo TCS3472 provee una salida digital de los valores percibidos de luz rojo, verde, azul (RGB) y despejado o sin filtro. Un filtro infrarrojo, integrado en el chip y localizado sobre los fotodiodos, minimiza la componente espectral infrarroja de la luz entrante y permite medir las componentes de color de una forma precisa. La alta sensibilidad, en rango amplio dinámico y el filtro infrarrojo hacen del TCS3472 una solución ideal para medición de color para su uso bajo unas condiciones variables de iluminación y a

través de materiales que la atenúen. El sensor tiene una gran variedad de aplicaciones incluyendo el control RGB de la iluminación LED, iluminación de estado sólido, productos de salud y deporte, control de procesos industriales y diagnóstico médico. Adicionalmente, el filtro IR permite que el TCS3472 se pueda utilizar para medición de luz ambiental."

Figura 21 – *TCS34725*

SENSOR DE COLOR RGB TCS34725 CON FILTRO IR ARDUINO

El TCS34725 es posiblemente el mejor sensor de color del mercado, incluye sensores RGB y de luz blanca. Cuenta con filtro bloqueador de IR integrado, que minimiza el espectro IR logrando unas medidas de color muy precisas.

Esto hace que las lecturas correspondan a color "real" o visible dado que los humanos no vemos el espectro IR. Además cuenta con un rango dinamico 3,800,000:1 con ajuste de ganancia automatico por lo que se puede incluso usar detras de un cristal oscuro.

Incluye un regulador de voltaje por lo que se puede usar tanto a 3.3V como a 5V, incluye un led neutral de 4150ºK controlado por un driver Mosfet para iluminar la superficie a medir que puede encenderse y apagarse muy facilmente con cualquier salida logica.

Se puede usar con cualquier microcontrolador con pines I2C, se conecta el pin VDD a la salida de 3 o 5V DC, la masa al pin GND, el SCL al I2C Clock y el pin SDA al de datos I2C.

- Peso: 3.23g
- Dimensiones: 20.44x20.28mm
- Este chip usa la direccion I2C 7-bit 0x29.

Ficha Tecnica TCS34725

Tutorial Arduino Sensor de Color TCS34725 Arduino

Codigo Phyton TCS34725 para Raspberry Pi

Precio : 8,75 €

PVP (IVA incluido) : 10,59 €

Figura 22 – *Precio y características del producto TCS34725 en la tienda de electrónica online CETRONIC [29]*

Una de las ventajas que disponen estos sensores a la hora de realizar las mediciones de la luz incidente es que, gracias a su diseño de 4 canales (rojo, verde, azul y sin filtrar) permiten la medición de las componentes de la luz, ampliando las posibilidades de cara a la realización de algoritmos de balance de blancos basados en las componentes en color de la imagen.

Adicionalmente, a partir de las componentes en color medidas, se pueden establecer relaciones con los valores del espacio de color del CIE1931, así como con la temperatura de color de la fuente de iluminación a través de tales coordenadas.

3.3. Sistema de procesamiento

El sistema de procesamiento estará compuesto por dos componentes fundamentales. Por un lado se tendrá la unidad de pre-procesamiento que será la encargada de la adecuación de la señal que proviene del sensor de forma que la unidad de procesamiento propiamente dicha, que será el segundo componente, pueda utilizar tales datos con el fin perseguido en el presente trabajo.

Unidad de pre-procesamiento

La unidad de pre-procesamiento estará compuesta por un micro-procesador Arduino Mega2560 rev3, siendo su función principal la de capturar los datos del sensor y enviar tales datos a través de comunicación serie con el procesador principal. La descripción del código que va a ser ejecutado por el micro-controlador puede ser revisada en el **Anexo 8**. El conexionado entre el sensor y la placa Arduino se realizará conforme al **Anexo 1**, mientras que la conexión del Arduino con el PC se realizará a través de cable USB, el cual permitirá las comunicaciones serie y será fuente de alimentación.

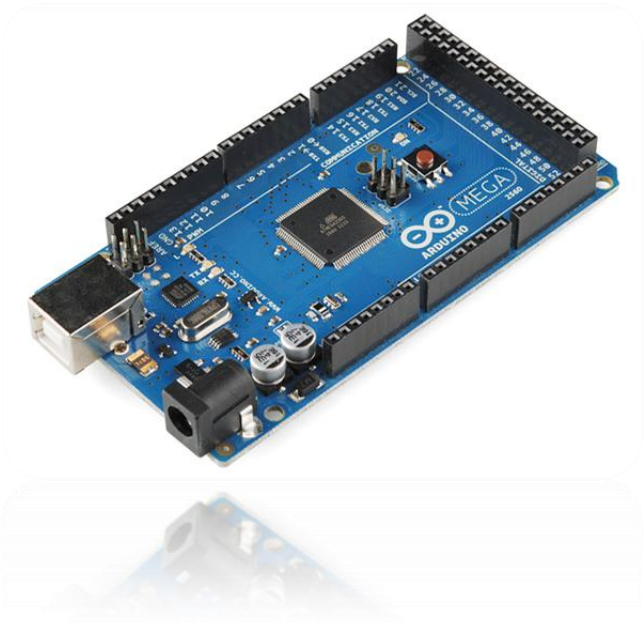


Figura 23 - *Placa Arduino Mega2560*

Unidad de procesamiento

La unidad de procesamiento que va a ser utilizada para el desarrollo del presente Trabajo se basa un PC con sistema operativo WINDOWS7 sobre el cual se ejecutará un programa sobre MATLAB. La estructura software que se ejecutará en el citado programa será la mostrada en la siguiente imagen.

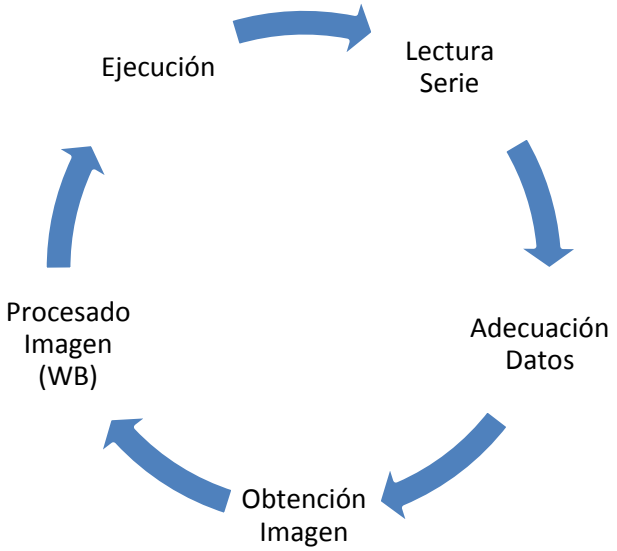


Figura 24 – *Estructura del programa*

- *Lectura Serie*: se leerán los datos que vienen de la unidad de procesamiento y que tienen que ver directamente con los datos obtenidos a través del sensor.
- *Adecuación de los datos*: los datos obtenidos del sensor tendrán que ser sometidos a un proceso de adecuación (establecido por la calibración del sensor) antes de poder ser utilizados.
- *Obtención Imagen*: se obtendrá la imagen que va a ser sometida al proceso de balance de blancos.
- *Procesado Imagen (WB)*: la imagen que fue obtenida en el apartado anterior es procesada a través de algoritmos de balance de blancos.

3.4. Resultado final del dispositivo

El resultado final de la conjunción de las partes descritas anteriormente se puede observar en las siguientes imágenes. Se basa en una estructura de madera sobre la que se ha instalado un punto de luz halógeno. A media altura dispone de una base para posar los filtros que van a ser utilizados para desviar la tonalidad de la fuente de color. En la base se instalará la placa Arduino con el sensor RGB para que pueda medir la luz incidente proveniente de la fuente.

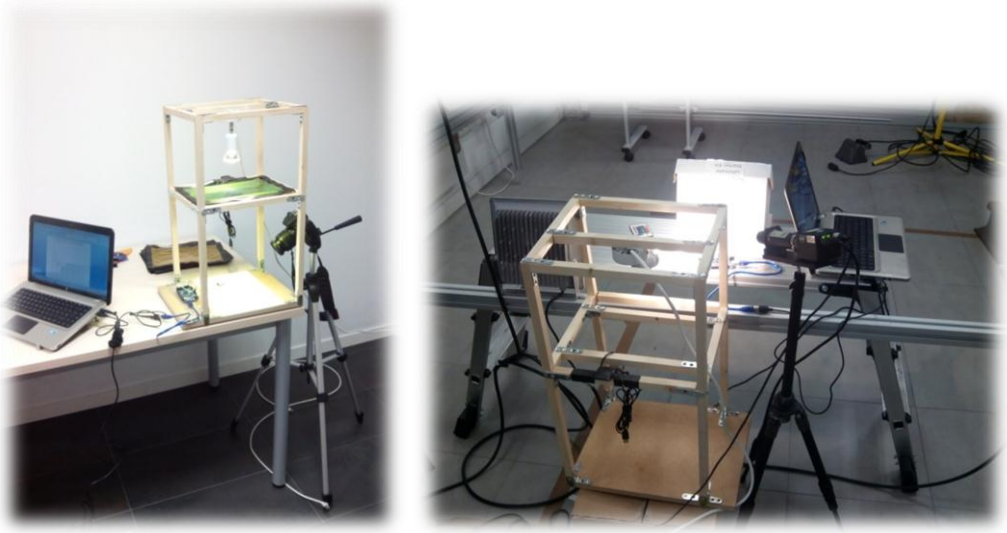


Figura 25 - *Prototipo*

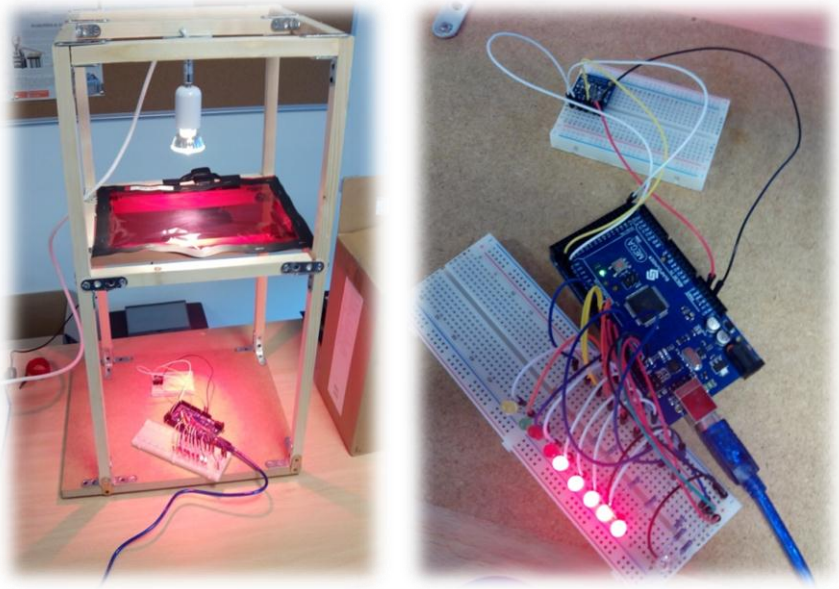


Figura 26 – *Detalle del soporte de filtros y base*

La siguiente imagen muestra los filtros de color plásticos que son utilizados.

Figura 27 – *Filtros utilizados*

3.5. Desarrollo experimental: calibración del sensor

El desarrollo experimental marcará la metodología que va a ser seguida de cara a la comprobación del buen funcionamiento del sensor integrado en el diseño mostrado en el apartado anterior. En el caso que compete a este trabajo se han realizado dos tipos de desarrollos experimentales incrementales, comenzando por la comprobación del sensor en crudo (sin realizar una calibración) y calibrando el mismo.

En cualquier caso la base de los ensayos es común y se fundamenta en la capacidad que tiene el sensor para percibir la componente inducida por la fuente de luz sometida a la acción de diferentes filtros.

La respuesta del citado sensor resultará crítica en la cadena de ajuste de imágenes con el concepto de balance de blancos que es el que más impronta tiene para el presente trabajo.

Los ensayos son de tipo incremental debido a que los valores obtenidos en el primer ensayo con el sensor sin haber sido calibrado servirán como base a la siguiente etapa en la cual se entrenará el sensor sobre unas respuestas conocidas.

3.5.1. Estructura software de análisis

El desarrollo de esta parte se realiza a través de MATLAB como soporte software. Como forma de comprobación de la veracidad que otorga el dispositivo se pueden establecer dos formas, por un lado una

forma subjetiva que está basada sobre la percepción de nuestros ojos sobre una superficie en la zona objetivo de color blanco mientras que por otro lado se puede considerar un ensayo objetivo como aquel que permite una mayor reproducibilidad de los resultados, usando para este último caso la medida otorgada por una cámara sobre la misma superficie. En el caso que compete, los resultados que van a ser analizados y mostrados serán aquellos procedentes del método objetivo.

La parte fundamental en este aspecto es la de la adquisición de los datos provenientes del sensor así como aquellos que se recogen a través de la cámara sobre las superficies consideradas como "tester".

El objetivo final será el de completar una tabla de la forma mostrada a continuación donde "sen" significa procedentes del sensor mientras que "test" serán provenientes de la superficie de "tester" a través de la cámara:

Tabla 2 – *Tabla a rellenar*

Rsen	Gsen	Bsen	Rtest	Gtest	Btest
…	…	…	…	…	…

Debe quedar claro que los valores que van a ser obtenidos de las superficies denominadas "tester" no van a provenir de las medidas en los canales de un único pixel, sino que se realizará una media sobre un área determinada para obtener unos valores más robustos.

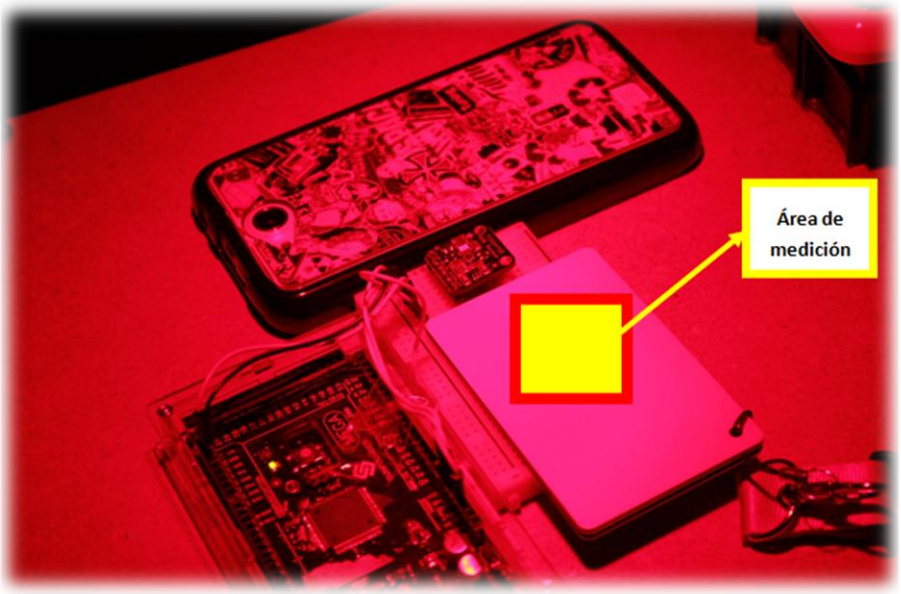


Figura 28 – *Muestra del área de medición sobre la superficie tester*

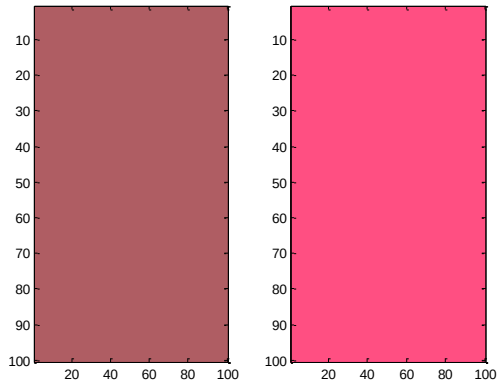
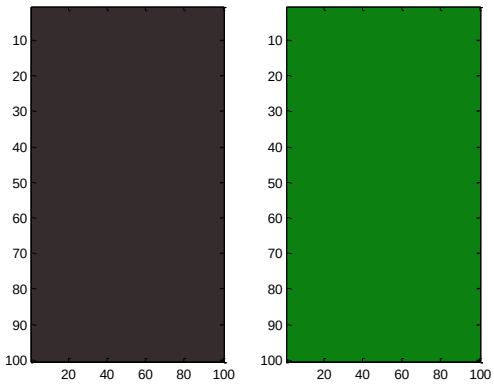
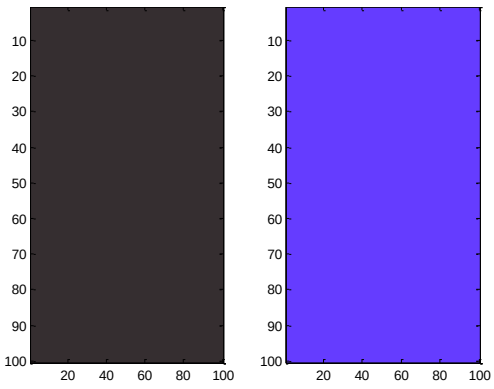
Los datos que han sido obtenidos que tienen correspondencia entre los datos capturados por el sensor y los obtenidos a través del procesamiento de la superficie tester de la cámara pueden ser consultados en los **Anexos 2, 3 y 4**.

3.5.2. Ensayos sin calibrar

En una primera fase se probó el sensor con diferentes filtros, asignando la salida proporcionada directamente por el mismo como el color medido. A la vez se recogió la media del color en la superficie blanca del "tester" para la misma fuente de luz con ayuda de una cámara.

Las siguientes imágenes muestran las diferencias, para casos de filtros rojo, verde y azul, entre los valores obtenidos.

Tabla 3 – *Valores sin calibrar para rojo, verde y azul (izquierda valores del sensor, derecha valores de la cámara)*

	
R = 175 G = 93 B = 99	R = 255 G = 79 B = 130
	
R = 52 G = 44 B = 45	R = 12 G = 129 B = 17
	
R = 53 G = 46 B = 48	R = 101 G = 60 B = 255

Como era de esperar la calibración del sensor dista mucho de ser la óptima cuando se trata de medir las características de la iluminación incidente, debido a que claramente se puede ver el predominio de la sensibilidad sobre el canal rojo, mientras que la sensibilidad de verde y azul distan mucho de esta primera, apareciendo tonos grises en la respuesta.

Este sensor, a pesar de ser útil para medir las características de la luz, está diseñado más específicamente para la medición de colores en la proximidad, haciendo uso de un LED que dispone sobre el mismo circuito. En cualquier caso el uso de un ajuste para calibrar el sensor puede proveer de una respuesta mucho más próxima a la realidad.

3.5.3. Ensayos calibrando el sensor

De cara a obtener unas características fehacientes de la fuente de luz incidente, se deberá llevar a cabo un proceso de calibración del sensor, que permita ajustar los valores provenientes del sensor a unos valores que se correspondan con la realidad. En este proceso se generará una función que tomando como entrada los valores en crudo del sensor, devuelva otros que se ajusten mejor a aquellos recogidos por la cámara sobre la superficie denominada "tester" que son considerados como buenos.

Para este objetivo se utilizarán cuatro métodos diferentes de ajuste de forma que el análisis de los datos se haga de forma incremental en complejidad así como que a la vez permita la comprobación de varias técnicas con el objeto de ver cuál puede conseguir un mejor ajuste. Los métodos propuestos son los siguientes:

- Regresión lineal mono-variable.
- Regresión lineal multi-variable.
- Regresión por Splines con algoritmo MARS.
- Redes neuronales.

Adicionalmente, como el conjunto de datos disponibles para realizar el entrenamiento de los antedichos métodos de machine learning, es muy escaso, se utilizarán mecanismos de "cross-validation". El conjunto total de datos de partida se dividirá en cuatro bloques de igual tamaño (si es posible) y se utilizarán tres de ellos como conjuntos de entrenamiento quedando el cuarto como conjunto de validación o test, intercambiando el grupo que se utiliza como test en cuatro iteraciones diferentes del algoritmo (esta característica no aplicará a las redes neuronales, debido a que el procedimiento de obtención es automático).

Para la comprobación de qué método de entre los cuatro preseleccionados (junto a la combinación de características) son las que mejor ajustan el comportamiento real de la luz incidente, se tomará la media de la suma de los errores cuadráticos medios de los cuatro ensayos (ensayos derivados de combinaciones provenientes de la utilizción de "cross-validation").

Cabe destacar que las medidas que se tomen como reales en el montaje propuesto en realidad no reproducirán fehacientemente el color real de la fuente, sino que pasarán por un proceso de conversión a través de los sensores de una cámara digital que introduce cierto error en la forma que procesa el color. La consecuencia directa de esto es que el ajuste que se consiga con la **calibración estará determinado para una cámara específica** (capacidad de percibir el color por su sensor) sin conseguirse una generalización. Para conseguir una generalización, las pruebas de calibración no deberían hacerse con una cámara tomando capturas de una superficie blanca ("tester"), sino que debería utilizarse una cámara espectográfica que permita capturar de forma fiel las componentes de la luz emitidas por la fuente.

3.5.4. Métodos de calibración

Regresión lineal uni-variable

Como primera aproximación al establecimiento de una relación directa entre las características de la fuente de luz incidente sobre el sensor y las medidas a través de una cámara se utilizará la regresión lineal uni-variable debido a su sencillez.

Con los datos obtenidos (mostrados anteriormente) el proceso será el siguiente: por un lado se tendrán los valores correspondientes a los canales verde, rojo y azul provenientes del sensor y por otro lado se tienen aquellos medidos sobre la superficie denominada "tester".

A través de regresiones uni-variable entre canales respectivos (rojo sensor → rojo en "tester", verde sensor → verde en "tester" y azul en sensor → azul en "tester") se intentará establecer el patrón de la respuesta en función de la entrada, es decir, predecir la salida de color que se tendría (lo que se mide sobre el "tester") a través de los datos recogidos por el sensor.

Para este fin, se ha utilizado una plantilla de Microsoft Excel y se ha probado con correlaciones lineales, y adicionalmente logarítmicas, potenciales y exponenciales en los casos que haya sido posible.

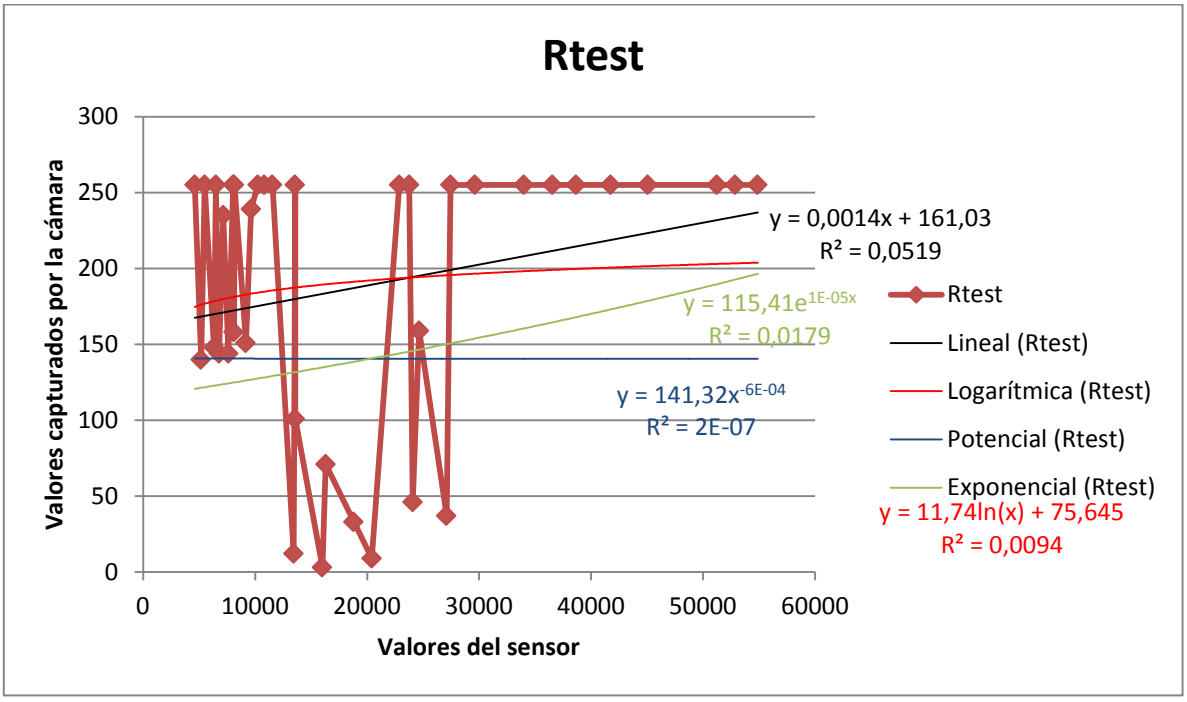


Figura 29 – *Modelo de regresión del canal rojo*

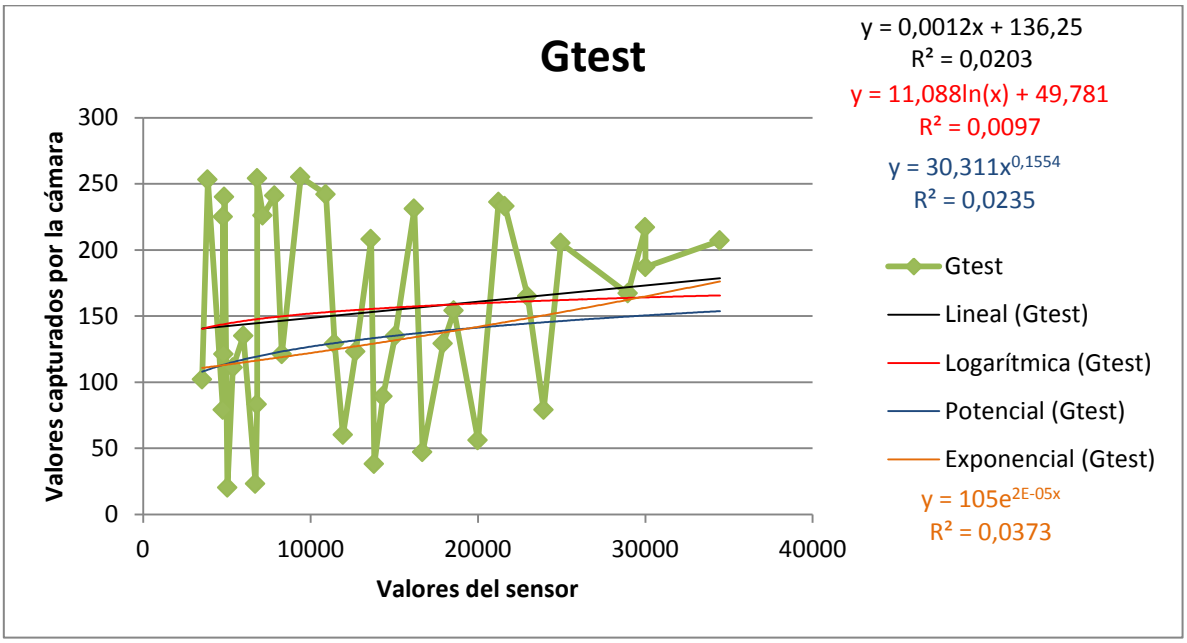


Figura 30 – *Modelo de regression del canal verde*

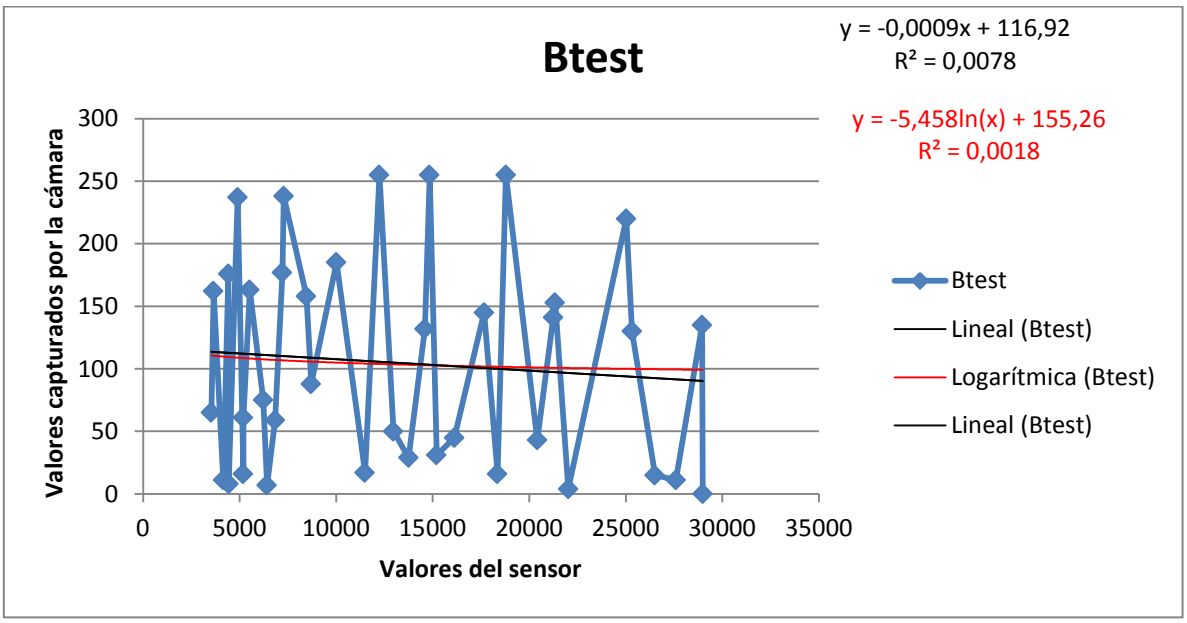


Figura 31 – *Modelo de regresión del canal azul*

A la vista de la dispersión de los puntos existentes y sobre todo del valor del coeficiente de determinación (R^2) en las regresiones se concluye que será muy complicado establecer una regresión uni-variable (lineal, logarítmica, exponencial) como modelo de predicción de los valores reales en función de las entradas establecidas por los valores provenientes del sensor.

Es por ello que se hace necesario buscar algún tipo de relación adicional que permita el establecimiento de más de una variable de las características medidas por el sensor para poder dar mayor robustez a la interpolación buscada. De tal forma el siguiente análisis será realizado con regresiones lineales multi-variable así mismo que con Splines y adicionalmente redes neuronales.

Regresión lineal multi-variable

En vez de ajustar con regresiones lineales simples, lo que se puede hacer es extender el modelo de regresión de forma que pueda acomodarse a un mayor número de variables[27].

Ecuación 3 – *Forma de la regresión lineal multi-variable*

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

Donde X_1 representa el j-ésimo predictor y β_1 cuantifica la asociación entre la variable y la respuesta. Se interpreta β_1 como la media del efecto en Y del incremento de una unidad en X_j, manteniendo el resto de predictores fijados.

El análisis que ha sido realizado es idéntico tanto para las regresiones lineales multi-variable como para las Splines. Se han elegido unas posibles combinaciones de variables que pueden ser relevantes para la representación del color. En este caso las variables obtenidas a través del sensor que conformarán los conjuntos de comprobación serán los canales rojo, verde, azul, claro y la temperatura de color (que es provista a través de una relación matemática por la propia librería de funciones del sensor).

Estas variables elegidas, serán utilizadas para relacionar cada uno de los colores RGB que se corresponden con el objetivo del presente trabajo. Para ello, se probarán diferentes distribuciones de variables de entrada para la regresión a ver cuál de ellas da un mejor resultado, se puede consultar en el **Anexo 17.**

Tanto en el caso de las regresiones lineales multi-variable como en las Splines, se realizarán regresiones de canales por separado, es decir, existirá una relación por cada canal, y como hay tres canales (Rojo, Verde y Azul); habrá tres regresiones por modelo probado.

Se utilizará el método de validación cruzada para obtener aquella relación que permita minimizar la media de la suma de errores cuadráticos (columna "Error") de la regresión con respecto la curva real para los conjuntos de test. Una vez obtenido aquel que minimiza tal error, se utilizarán todas las imágenes como datos de entrenamiento, para así robustecer el algoritmo que se obtenga con mayor número de datos de partida, que son bastante escasos (comparando los métodos de regresión lineal multi-variable así como las splines).

Las siguientes tablas muestran:

- Las cinco primeras columnas son la información que puede ser extraída de la librería del sensor que puede tener relevancia para la regresión. Las variables que son seleccionadas en cada ensayo como entradas (inputs) para el algoritmo de regresión, son resaltadas a través de un sombreado oscuro de la casilla correspondiente
- La sexta columna muestra la media de errores cuadráticos medios de cada uno de los ensayos.
- Cada fila representa una combinación diferente, ensayo, de variables de entrada en el algoritmo de regresión.

Tabla 4 – *Canal rojo con regresiones multi-variable*

ROJO	VERDE	AZUL	CLARO	CCT	Error
					11278,905
					16524,394
					16452,987
					11451,619

Tabla 5 – *Canal verde con regresiones multi-variable*

ROJO	VERDE	AZUL	CLARO	CCT	Error
					19201,642
					17265,152
					19146,989
					25636,617

Tabla 6 – *Canal azul con regresiones multi-variable*

ROJO	VERDE	AZUL	CLARO	CCT	Error
					20534,623
					32523,600
					30193,035
					30427,768

Splines

Por otro lado un algoritmo que puede ser de gran utilidad a la hora de establecer regresiones es el Multivariate Adaptive Regression Splines (MARS), el cual es una forma de análisis de regresión introducido por Jerome H. Friedman en 1991. Es una técnica de regresión no paramétrica que se puede ver como la extensión de los modelos lineales que automáticamente modelan no linealidades e interacciones entre variables.

Como se puede ver en las siguientes imágenes que sirven para ejemplificar el funcionamiento de este algoritmo antedicho, el ajuste a los datos utilizando es te tipo de algoritmos es mucho mejor cuando se presentan no linealidades dentro de un conjunto de datos que quiere ser predicho.

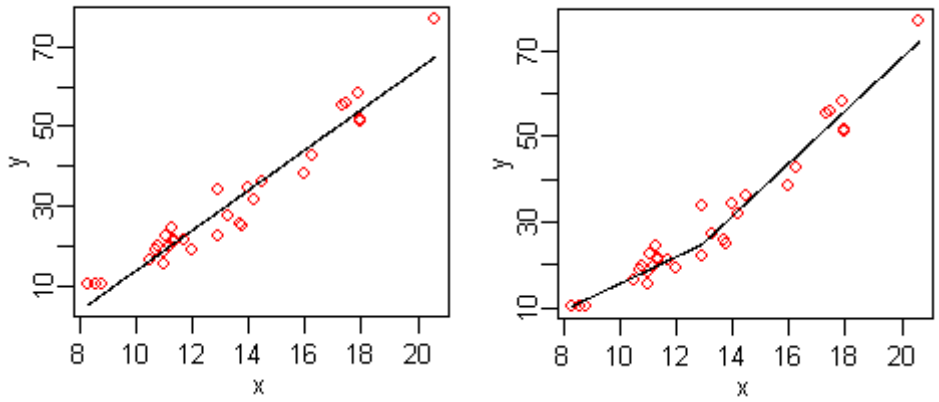


Figura 32 – *MARS tipo*

Otra de las ventajas que se pueden conseguir gracias al uso de este algoritmo es el descubrimiento de relaciones con múltiples variables independientes. Gracias a esto, se podrán introducir en el cálculo del modelo del canal de salida otras variables que en el caso anterior no eran introducidas por imposibilidad de aplicación de los modelos de regresión.

MARS construye sus modelos mediante la forma:

Ecuación 4 – *Forma de construcción de modelos en MARS*

$$\hat{f}(x) = \sum_{i=1}^{k} c_i B_i(x)$$

El modelo es un suma de funciones básicas Bi(x), siendo "ci" un coeficiente constante. Cada una de las funciones básicas puede tener una de las tres siguientes formas:

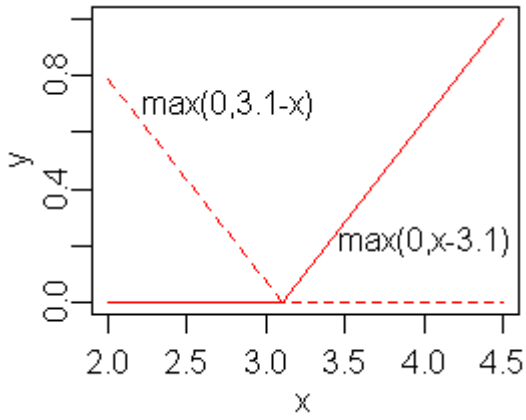
- una constante igual a 1.
- una función bisagra. Esta función tiene la forma max(0,x-constante) o max(0,constante-x). MARS automáticamente selecciona las variables y los valores para esas variables para los nudos de estas funciones bisagra.
- Un producto de dos o más funciones bisagra. Estas funciones base pueden modelar la interacción entre dos o más variables.

Como se ha comentado anteriormente, una de las bases de este algoritmo es el uso de funciones bisagra, y es por ello que será importante introducir un poco más a fondo de qué se trata el concepto de funciones bisagra. La forma de tales funciones es la siguiente:

Ecuación 5 – *Forma de funciones bisagra*

$$\max(0, x - c) \quad \text{o} \quad \max(0, c - x)$$

Donde C es una constante, denominada nudo. Una función bisagra tiene un valor de cero para parte de su rango, pudiendo ser utilizada para partir los datos en regiones separadas, cada una de las cuales se puede tratar de forma independiente.



$$6.1 \max(0, x - 13) - 3.1 \max(0, 13 - x)$$

Figura 33 – *Función bisagra*

MARS construye los modelos en dos fases, hacia adelante y hacia atrás. En la fase hacia adelante empieza con un modelo que consiste sólo en la media de los valores de respuesta, y repetidamente añade funciones base en parejas al modelo. En cada paso encuentra el par de funciones base que dan el máximo de reducción en la suma de errores cuadráticos. En la fase hacia atrás se intenta corregir el modelo sobre-ajustado que se ha generado en la anterior fase. Elimina los términos de uno en uno, de forma que se borren los menos efectivos en cada paso hasta que encuentra el mejor sub-modelo. La forma de comparación de los sub-modelos utiliza Validaciones Cruzadas Generalizadas (Generalized Cross Validation GCV). La mayor ventaja que aporta esta fase para la consecución de un mejor modelo es que en cada paso se puede elegir cualquier término para ser borrado, mientras que en la fase de avance hacia adelante en cada paso sólo se pueden ver la siguiente pareja de términos; gracias a esto típicamente la pasada hacia atrás descarta uno de las partes de la pareja y por tanto en el resultado final normalmente no aparecen parejas en las funciones bisagra.

Las restricciones que pueden acontecer sobre este algoritmo son:

- El usuario puede especificar el número máximo de términos en la pasada hacia adelante.
- También se puede establecer como restricción el máximo grado de interacción entre variables. Típicamente este grado se fija a uno o dos grados de interacción permitidos, pero mayores grados se pueden establecer siempre y cuando los datos lo garanticen.
- Se podrían establecer otras restricciones, como la especificación de que ciertas interacciones sólo puedan tener lugar entre unas variables predichas. Estas restricciones en todo caso tienen sentido cuando se tienen un conocimiento amplio del proceso que involucran los datos.

Los resultados según la combinación de variables de entrada así como los valores de la media de la suma de errores cuadráticos medios en el conjunto de test son reflejados en las siguientes imágenes.

Las siguientes tablas muestran:

- Las cinco primeras columnas son la información que puede ser extraída de la librería del sensor que puede tener relevancia para la regresión. Las variables que son seleccionadas en cada ensayo como entradas (inputs) para el algoritmo de regresión, son resaltadas a través de un sombreado oscuro de la casilla correspondiente
- La sexta columna muestra la media de errores cuadráticos medios de cada uno de los ensayos.
- Cada fila representa una combinación diferente, ensayo, de variables de entrada en el algoritmo de regresión.

Tabla 7 – *Canal rojo con Splines*

ROJO	VERDE	AZUL	CLARO	CCT	Error
					7581,015
					15030,093
					20314,277
					10200,695

Tabla 8 – *Canal verde con Splines*

ROJO	VERDE	AZUL	CLARO	CCT	Error
					14089,665
					16308,997
					17872,229
					30390,678

Tabla 9 – *Canal azul con Splines*

ROJO	VERDE	AZUL	CLARO	CCT	Error
					22891,387
					24402,459
					26985,036
					25833,128

Mejor solución

Una vez que se han realizado las pruebas con los diferentes métodos y con las diferentes variables y sus combinaciones sólo queda comprobar cuál es el que mejor aproxima la solución. Habrá que buscar el mejor método por canal de color que se quiere estimar. Una vez determinado (ver la siguiente tabla en la que se muestra el resultado ordenado en función de la suma de errores cuadráticos), se realizará la estimación completa considerando el conjunto completo de imágenes como datos de entrenamiento para conseguir robustecer el algoritmo utilizado. La siguiente imagen muestra los canales ordenados en función de la citada media de la suma de errores cuadráticos con respecto los conjuntos de test.

Tabla 10 – *Ensayos ordenados*

#	Suma Errores cuadráticos	Salida	Rojo	Verde	Azul	Claro	CCT	Método
1	7581,016	Rojo	■	■	■	■	■	Spline
2	10200,695	Rojo	■			■	■	Spline
3	11278,906	Rojo	■	■	■	■	■	Reg. Multi-variable
4	11451,619	Rojo	■			■	■	Reg. Multi-variable
5	14089,665	Verde	■	■	■	■	■	Spline
6	15030,094	Rojo	■	■	■			Spline
7	16308,997	Verde	■	■	■			Spline
8	16452,987	Rojo	■	■	■	■		Reg. Multi-variable
9	16524,395	Rojo	■	■	■			Reg. Multi-variable
10	17265,152	Verde	■	■	■			Reg. Multi-variable
11	17872,230	Verde	■	■	■	■		Spline
12	19146,989	Verde	■	■	■	■		Reg. Multi-variable
13	19201,643	Verde	■	■	■	■	■	Reg. Multi-variable
14	20314,278	Rojo	■	■	■	■		Spline
15	20534,624	Azul	■	■	■	■	■	Reg. Multi-variable
16	22891,388	Azul	■	■	■	■	■	Spline
17	24402,459	Azul	■	■	■			Spline
18	25636,617	Verde		■		■	■	Reg. Multi-variable
19	25833,128	Azul		■		■	■	Spline
20	26985,036	Azul	■	■	■	■		Spline
21	30193,035	Azul	■	■	■	■		Reg. Multi-variable
22	30390,678	Verde		■		■	■	Spline
23	30427,768	Azul			■	■	■	Reg. Multi-variable
24	32523,600	Azul	■	■	■			Reg. Multi-variable

Como anteriormente se había comentado, lo que se pretende con este análisis es conocer la mejor solución y combinación por cada canal de la respuesta esperada (canales rojo, verde y azul), para de esta forma poder obtener las características de la función. Los resultados se muestran a continuación:

Canal rojo: <u>todas las variables SPLINES</u>

Canal verde: <u>todas las variables SPLINES</u>

Canal azul: <u>todas las variables regresión LINEAL MULTI-VARIABLE</u>

Gracias a este análisis y al script presentado en los **Anexos 17, 18 y 19**, se establece la relación necesaria para conseguir predecir el valor de la fuente. No obstante se hará el proceso con redes

neuronales para demostrar la sencillez de su uso así como la capacidad que tienen de ajustar la regresión aceptablemente.

Redes neuronales artificiales

Otra aproximación que se puede establecer para comprobar la capacidad de predecir la componente de color inducida por la fuente de iluminación es utilizando mecanismos de redes neuronales artificiales. En este caso se ha decidido separar este ajuste de los dos realizados con anterioridad para gestionar de forma automática los grupos de entrenamiento, validación y test.

"Las redes neuronales artificiales son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Se trata de un sistema de interconexión de neuronas que colaboran entre sí para producir un estímulo de salida."

Para la adecuación de las redes neuronales, se realizarán ensayos modificando ciertos dos parámetros, que serán:

- El **número de neuronas** en la capa oculta de la red. Incrementando el número de neuronas en esta capa hace que la red sea más potente, pero al mismo tiempo requiere de más computación y es más fácil que se produzca sobre-entrenamiento o sobre ajuste a los datos de entrenamiento. Las neuronas probadas en la capa oculta serán: 10, 15, 20, 25 y 30.
- El segundo parámetro que se puede modificar es la **función a ser usada en el entrenamiento**. En este caso dos posibilidades serán utilizadas:
    - Levenberg-Marquardt. Esta función es la más rápida de algoritmos de "backpropagation" en la toolbox, ajustando el peso y los valores de tendencia de acuerdo al modelo de optimización de Levenberg-Marquardt. Tiene el problema de que requiere más memoria que otros algoritmos. (trainlm)
    - Por otro lado se probará también con "trainbr" en la cual se actualizan los valores de peso y tendencia de acuerdo con el modelo de optimización de Levenber-Marquardt. Minimiza una combinación de errores cuadrados y pesos, y determina la combinación correcta de forma que la red se generalice bien. Este proceso se denomina regularización Bayesiana.

En el **Anexo 21** pueden ser consultadas las figuras que muestran las iteraciones que han sido realizadas así como los resultados obtenidos en cada una de ellas. Así mismo, sobre todos los modelos anteriormente calculados se realizarán varias pruebas empíricas (con nuevas imágenes) para ver cómo responde la misma ante la predicción de los colores reales inducidos por la fuente de luz. Con los resultados obtenidos de estas pruebas se seleccionará la combinación de neuronas en la capa oculta así como la función usada en el entrenamiento más adecuada.

Las imágenes que serán utilizadas en esta última correlación, así como los datos provenientes del sensor y de la cámara, serán las que aparecen en los **Anexos 5, 6 y 7**.

Selección

A la vista de los resultados obtenidos a través de las diferentes iteraciones del **Anexo 21** y del test realizado sobre el nuevo conjunto de tres imágenes, se seleccionará como óptimo el desarrollo representado por una red neuronal con **15 neuronas en la capa oculta y con una función Levenberg-Marquardt** por tener la menor desviación absoluta con respecto a las imágenes de test.

Tabla 11 – *NN15LM*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	255	37	90
Final3	172	246	49	112	230	33
Final4	7	139	255	26	114	255

Error = 261

Probando de la misma forma con las regresiones lineales y Splines que fueron calculadas y seleccionadas con anterioridad, se ve que el error cometido es superior en este segundo caso; siendo entonces más adecuado el uso de redes neuronales, como se había postulado en un primer momento.

Tabla 12 – *Splines y lineal multi-variable*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	245	47	46
Final3	172	246	49	105	229	51
Final4	7	139	255	13	113	201

Error = 341

4. Algoritmos de Balance de Blancos

Los algoritmos de balance de color o balance de blancos son una clase de algoritmos de procesamiento de imágenes que se efectúa para compensar los efectos de la fuente de luz sobre los objetos capturados, que sufren una predominancia de color determinada por tal fuente. De cara a evitar tal desplazamiento en los niveles de color, el algoritmo genérico de balance de blancos se basa en [14]:

- Estimación de la temperatura de color de la fuente de luz que ilumina la imagen.
- Ajuste de los contenidos de la imagen para eliminar el efecto de la fuente de luz.

El primer punto, que se basa en la medición de las características de la iluminación incidente se ha tratado en el apartado anterior, mientras en el presente punto se tratará en profundidad el ajuste de la imagen conocida la medición.

Por un lado cabe destacar que los algoritmos de balance de blancos tradicionales se catalogan en cuatro categorías diferentes [14], que son, método del Mundo Gris ("gray world method"), método Retinex, método de la Gama Tonal ("gamut mapping") y método de la reflexión especular ("specular reflection"). Aparte de éstos, existen gran número de métodos derivados como se explica en [10], donde existen además: mezclas de Mundo Gris y Retinex, Mundo Gris ponderado por la desviación típica, Parches blancos en el espacio YCbCr, etc.

Como anteriormente había sido mencionado, los sensores que caracterizan la fuente de iluminación normalmente no están disponibles debido al incremento de precio que supondría sobre los dispositivos de captura (cámaras). Por tal efecto los algoritmos de balance de blancos se fundamentan por norma general en cálculos estadísticos sobre las imágenes obtenidas, y por ello muchos métodos no son robustos en el sentido de que su rendimiento depende altamente en el contexto de la escena o la fuente de iluminación.

Con el objeto de evitar tal dependencia del citado contexto de la escena o de la fuente de iluminación sobre el rendimiento del algoritmo, habrá que adaptar los métodos existentes a la nueva información capturada a través del sensor RGB que permitirá caracterizar la fuente de iluminación.

Otro aspecto que se debe tener presente es el espacio de color utilizado para la ejecución del ajuste de la imagen. Como se detalla en [21], para el renderizado de fotografías es importante preservar los tonos grises a pesar de las variaciones en la iluminación ambiente. Cuando la iluminación es conocida, el balance de blancos para preservar tales tonos de gris se puede realizar en diferentes espacios de color. La conclusión obtenida al comparar cuatro espacios de color (XYZ, sRGB, Bradford, RGB del sensor de la cámara) para realizar dicha acción, se ha preferido utilizar XYZ y sRGB. En este caso se utilizará el espacio RGB.

4.1. Método del Mundo Gris y Retinex

Este método es uno de lo más simples existentes [28]. Se basa en el cálculo de un único parámetro estadístico de toda la imagen que se quiere renderizar, que es la media aritmética de los valores de los pixeles de la imagen, asumiendo como uniforme la región de interés. A pesar de la simplicidad que presenta este algoritmo, existen numerosas variaciones.

No indicado para:

Escenas que tienen amplias zonas de un color uniforme.

Por otra parte, el método Retinex o método del parche blanco se basa en tomar en cuenta el valor más grande en cada componente de color como representación del blanco de la imagen. Computacionalmente se calcula al encontrar la máxima intensidad en cada canal, dado por:

Ecuación 6 – *Cálculo computacional del algoritmo Retinex*

$$I_i = \max\{f_i(x, y)\}.$$

Como el dispositivo desarrollado permite medir las componentes de la fuente de iluminación del conjunto de la imagen capturada, la adaptación de este algoritmo puede hacerse de forma directa, estableciendo que el sensor es el parche blanco de la imagen. Es por ello que el algoritmo que será adaptado será el de Retinex.

No indicado para:

Escenas llenas de superficies abrasivas o sometidas a una fuente de luz saturada.

Escenas en las que la iluminación no es muy uniforme.

Como el blanco de una imagen idealmente está formado por la reflexión de todas las longitudes de onda incidentes sin absorber ninguna, se puede realizar un mapeado de la señal de entrada capturada por el sensor de la cámara de forma que se modifiquen los límites de máxima luminosidad correspondiente a cada canal, teniendo en cuenta que ningún valor de canal de la imagen será superior al dado por el sensor:

Los valores correspondientes por canal se calcularán a través de extrapolación:

Ecuación 7 – *Adaptación del algoritmo Retinex*

$$\text{Pixel red} = \text{PixelRED} * \frac{\text{sensorRED}}{\text{maxRED}}$$

$$\text{PixelGREEN} = \text{PixelGREEN} * \frac{\text{sensorGREEN}}{\text{maxGREEN}}$$

$$\text{PixelBLUE} = \text{PixelBLUE} * \frac{\text{sensorBLUE}}{\text{maxBLUE}}$$

Los resultados obtenidos al aplicar este ajuste pueden ser vistos en las siguientes imágenes, las cuales han sido tomadas a utilizando una plantilla de calibración de color que ha sido impresa (lo cual hace que no disponga de colores calibrados, pero que sirve de modelo subjetivo para conocer la mejoría que puede ser obtenida en las imágenes al utilizar el algoritmo de balance).

Figura 34 – *Plantilla Colorchecker*

Las siguientes imágenes muestran los resultados de aplicar el ajuste de blancos a las imágenes con la selección de la red neuronal de 15 neuronas en la capa oculta y con una función Levenberg-Marquardt. Se muestran dos imágenes por toma; la original (a la izquierda) y la ajustada (a la derecha). También se muestra el color capturado en cada toma. Las tomas se corresponden a una utilización sin filtros (luz halógena con tono amarillento), filtro rojo, verde y azul.

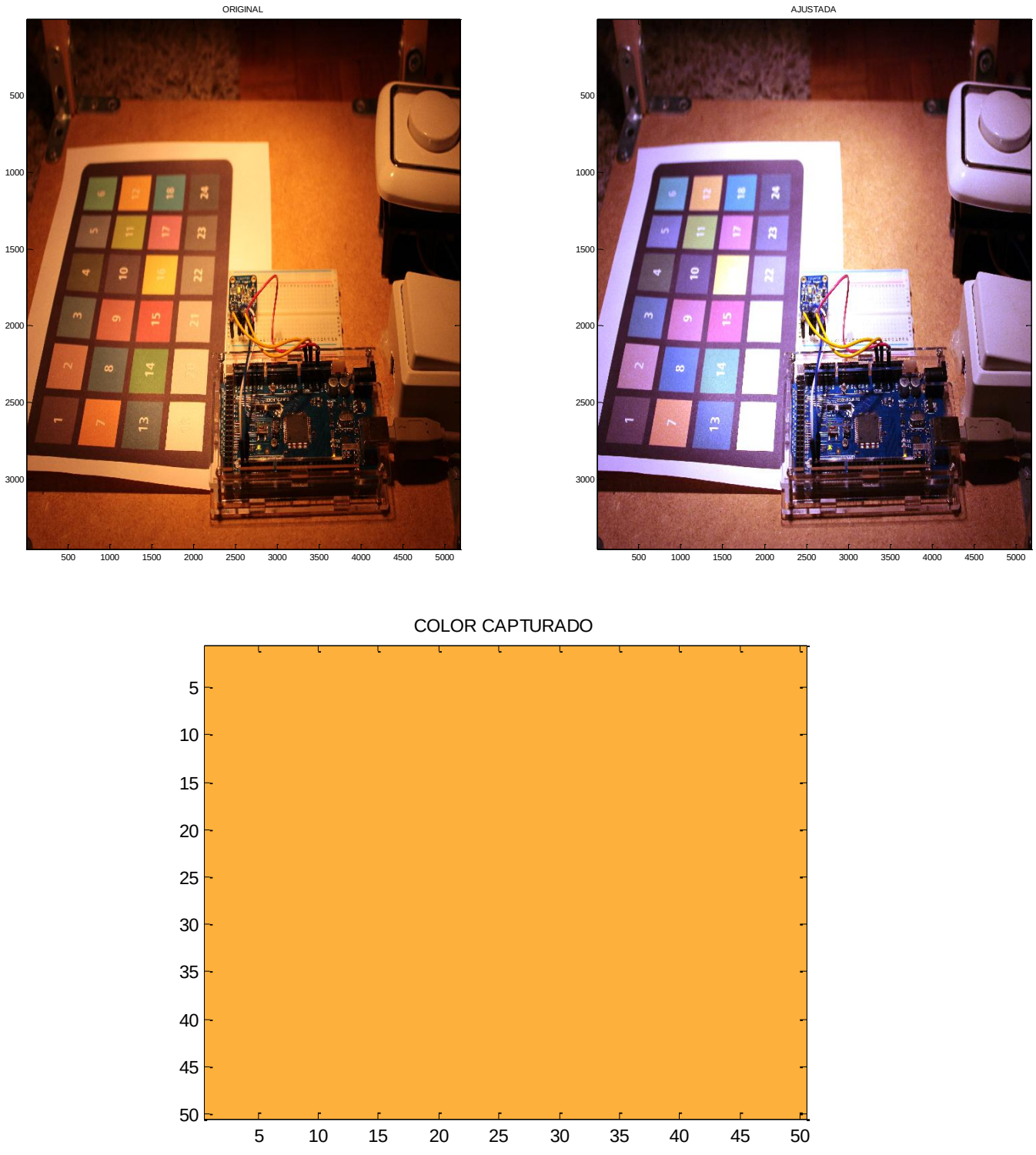


Figura 35 – *Sin filtro*

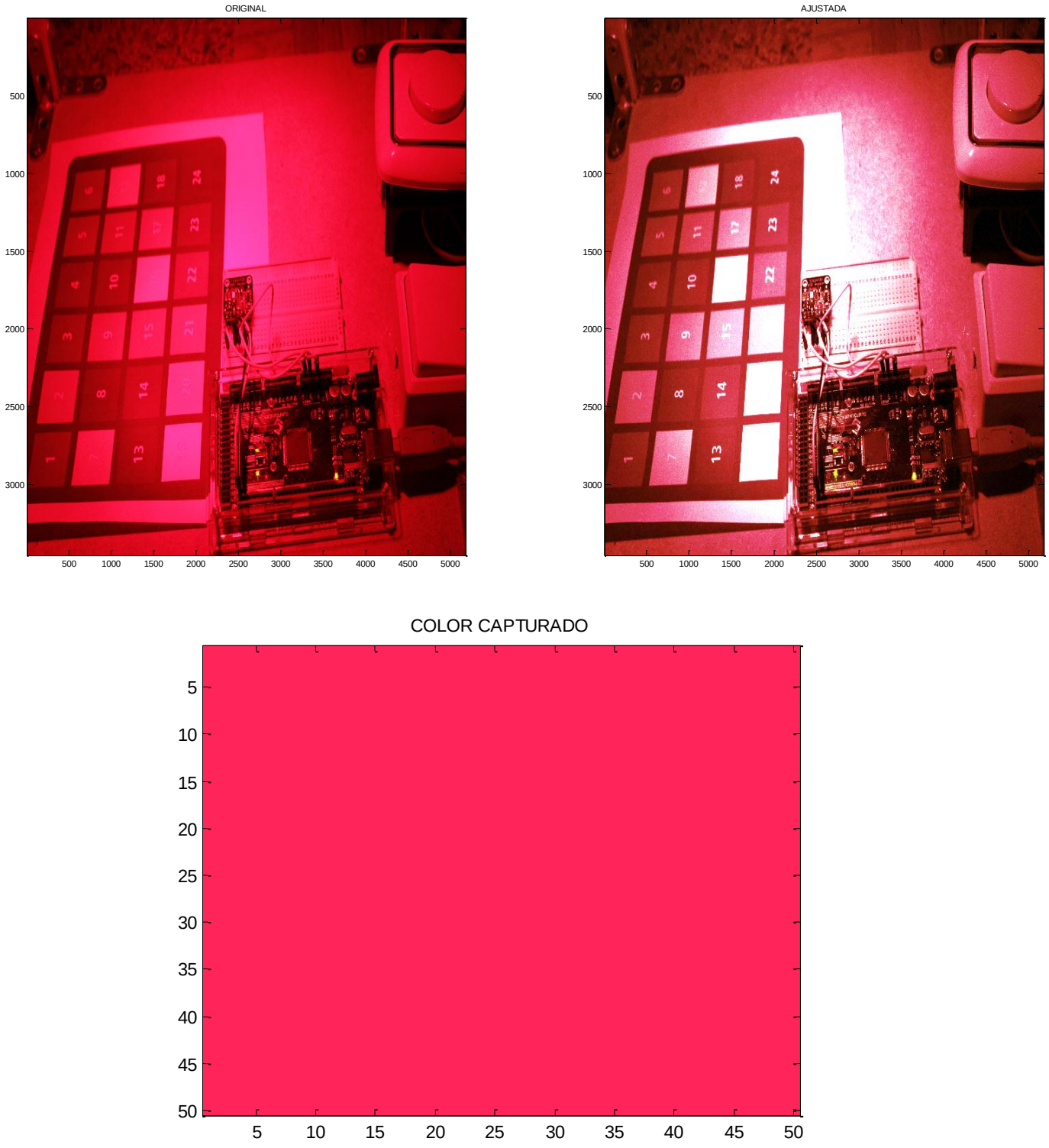


Figura 36 – *Filtro rojo*

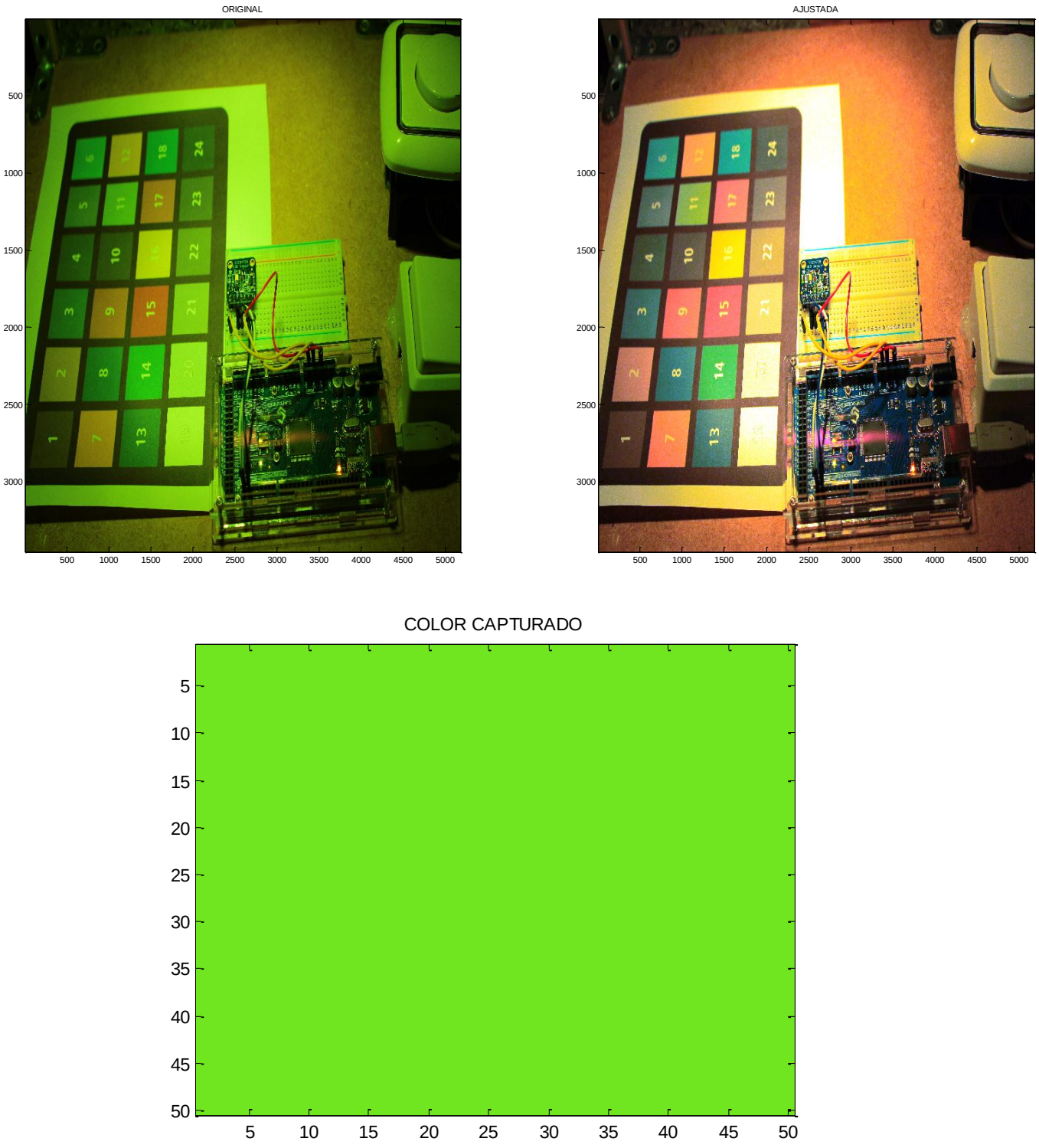


Figura 37 – *Filtro verde*

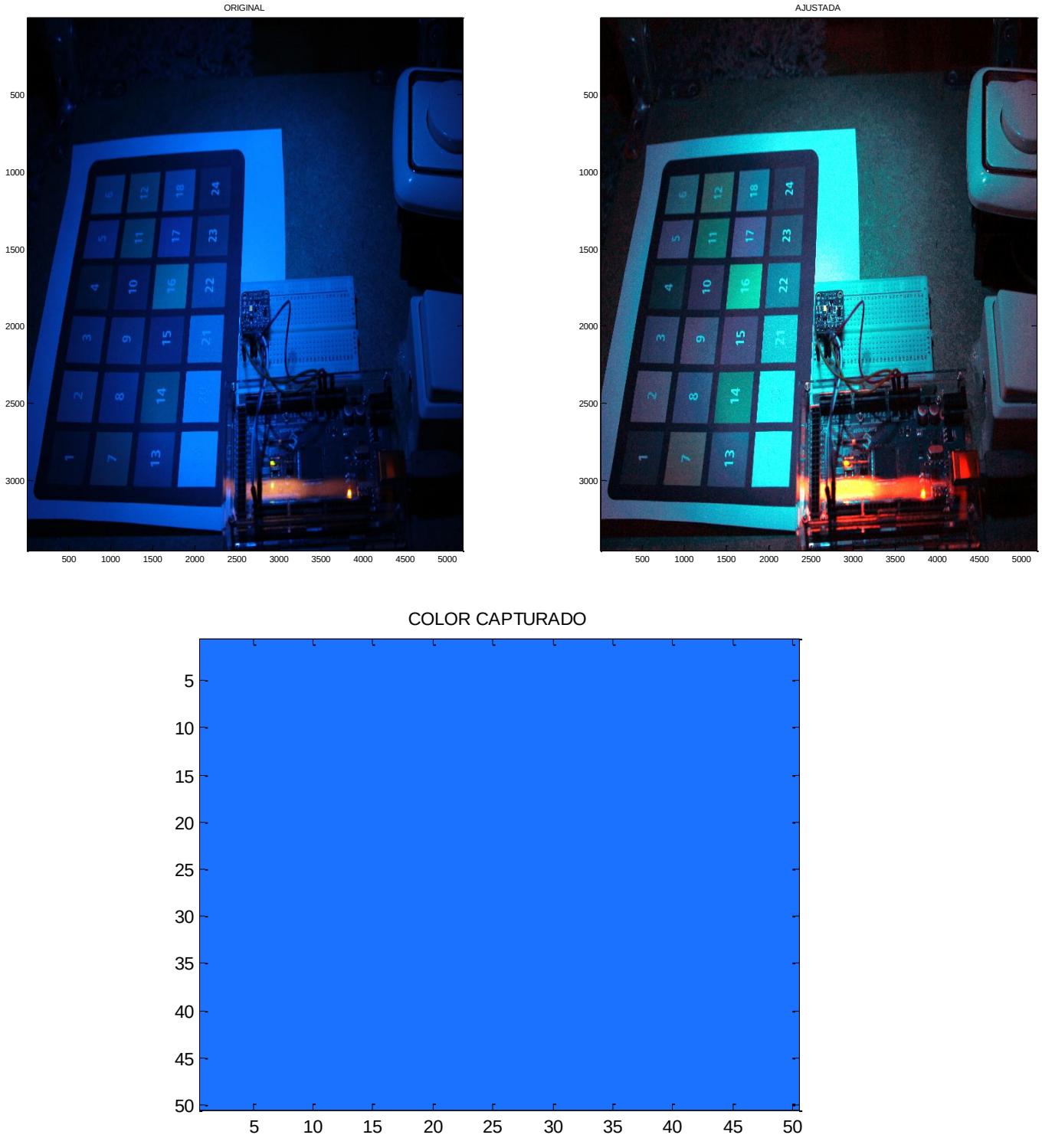


Figura 38 – *Filtro azul*

Como se ha podido observar, gracias a la utilización de la red neuronal y del algoritmo de balance de blancos se consigue una recuperación de colores bastante aceptable en todas las imágenes mostradas, siendo el blanco el color que mejor se ha conseguido ajustar, que era el objetivo desde el comienzo.

5. Resultados y conclusiones

La primera conclusión que puede ser extraída de los resultados obtenidos en el anterior apartado es que el dispositivo elegido como caracterizador de la fuente de iluminación **tiene usos potenciales para mejorar la calidad de las imágenes** tomadas en ambientes en los cuales la fuente de iluminación tiene una influencia extrema en la desviación del color respecto al que debiera ser bajo luz neutra. También debería por tanto ser válido para ambientes en los cuales la desviación no sea tan extrema.

No obstante, como se ha podido observar en el desarrollo del presente trabajo, el ajuste que puede ser conseguido a través de la red neuronal como estimador de los valores inducidos por la fuente de iluminación podría mejorarse. Esto se debe en su mayoría a que el **grupo de entrenamiento** es bastante **escaso**, cabe recordar que sólo se dispone de 60 elementos para hacer todo el análisis.

Una medida de mejora sobre la presente implementación se fundamentaría en hacer una **caracterización más amplia de dat**os. Consistiría en:

- Formar un conjunto de datos más amplio y heterogéneo.
- Utilizar un mayor número de filtros o fuentes capaces de generar mayor número de tonalidades. También probar en ambientes exteriores.
- Hacer la calibración con espectrómetros en vez de realizando medidas a través de cámaras fotográficas (estas últimas inducen error por el método de captura de la imagen, además de que producen ajuste para el modelo en concreto utilizado.).
- Hacer variaciones en la intensidad lumínica.
- Etc.

También se pueden observar en los resultados dos efectos. Por un lado el efecto halo que hace la fuente de iluminación elegida. El halógeno proyecta la luz, dejando en la zona central un **anillo de más intensidad que en los extremos**. Esto se debe entre otras cosas al ángulo de apertura de la bombilla, que al estar situada a una altura insuficiente del plano de recolección de datos, produce una iluminación no homogénea en intensidad. Otra parte se debe también a la heterogeneidad inherente a la fabricación de la bombilla y sus reflectores internos.

Figura 39 – *Efecto halo de la luz*

El segundo efecto es que se observa que en las imágenes en las cuales las **longitudes de onda son muy acotadas** (el caso del filtro rojo) los **colores apenas son recuperados**. Esto se debe, a como fue explicado en el capítulo de conceptos básicos "**¡Error! No se encuentra el origen de la referencia.** Formación del color", que si no se dispone de las longitudes de onda en la imagen original que conforman los colores de la misma, los objetos se apreciarán negros y como máximo se conseguirá un tono de gris más intenso como resultado del ajuste de balance de blancos.

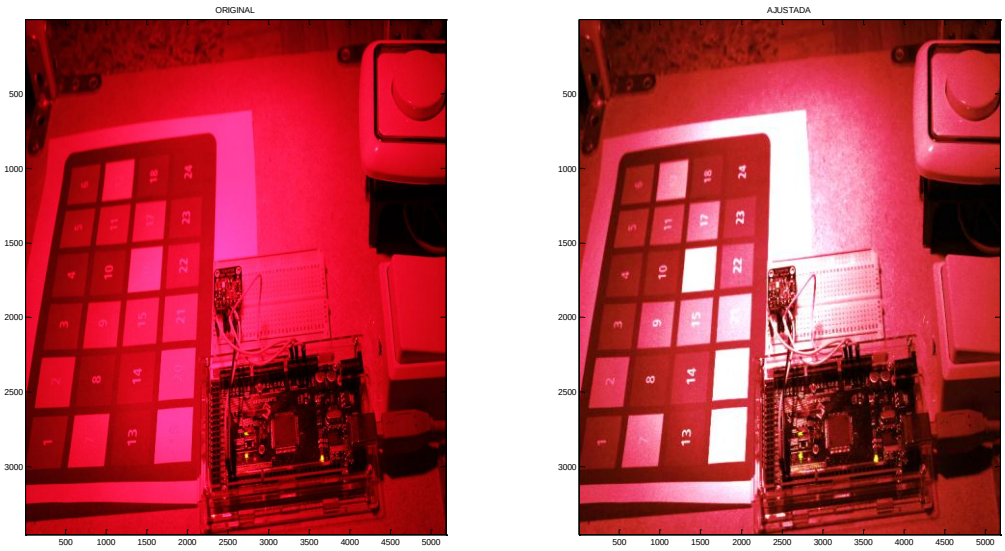


Figura 40 – *Imagen tomada con longitudes de onda muy acotadas, y por tanto con colores no recuperables. Los blancos sí se recuperan.*

De cualquier forma, los blancos se recuperan (mejor en la zona de iluminación más intensa y homogénea) y en ciertos casos también hay recuperación de colores.

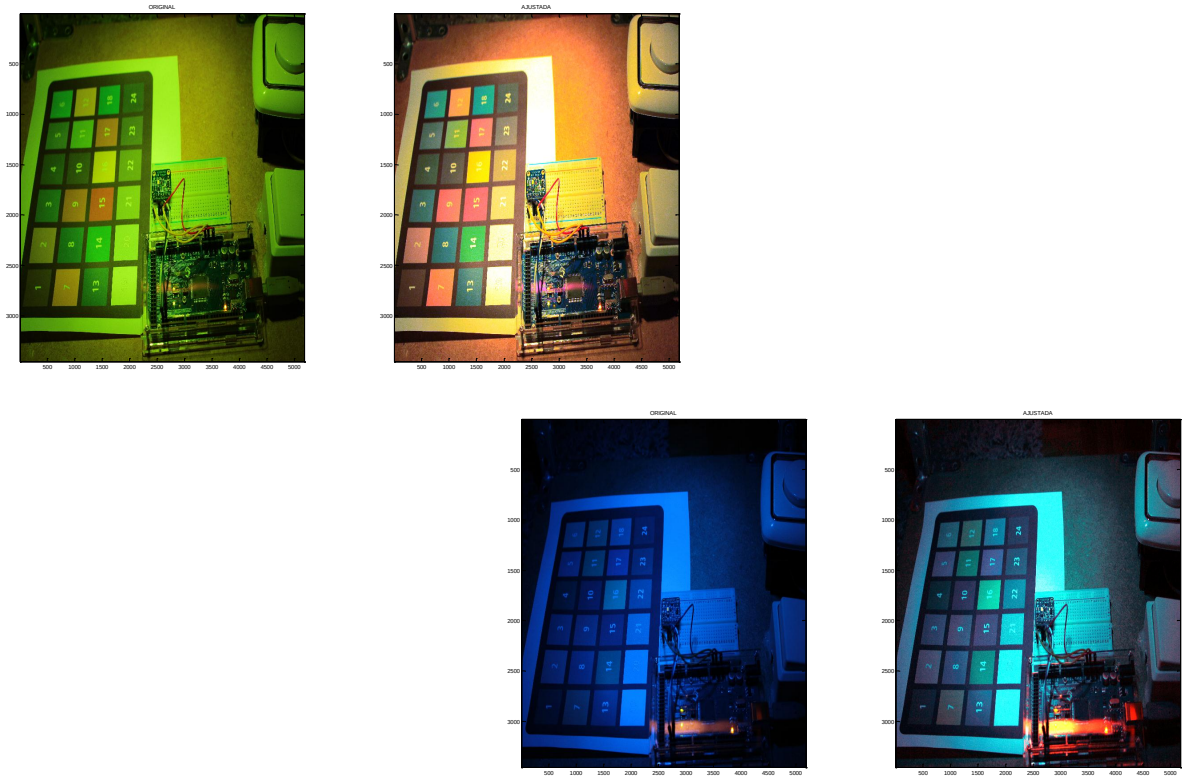


Figura 41 – *Ejemplos de recuperación de colores y de blancos (imagen original menos sesgada en longitudes de onda)*

Por otro lado, herramientas como **Photoshop** y otros tipos de software permiten realizar una corrección de las imágenes a través de valores de temperatura de color, lo cual facilita la inteligibilidad del método ya que la caracteriza la fuente de iluminación a través de un único parámetro. El problema se presenta en que las librerías proporcionadas por el sensor a través de las cuales puede obtenerse este parámetro, no son librerías calibradas ya que responden a relaciones matriciales a través de los valores en el espacio XYZ que captura el sensor sin calibrar.

Así mismo, la poca información existente acerca de cómo relacionar el espacio de color **RGB** con la **temperatura de color (CCT)**, sin tener que hacer transformaciones entre espacios de color que inducen bastante error, es otro factor limitante. En caso de que pudiera relacionarse el color en RGB con ese parámetro, se facilitaría enormemente la ejecución de la corrección de color. Como solución podría realizarse una red neuronal (teniendo en cuenta que deberíamos tener un número suficiente de datos) que relacione valores en RGB con el valor de la temperatura de color, para ser aplicados a los valores obtenidos una vez calibrados.

6. Referencias

[1]. Ricoh, Balance de Blancos. (accedida en diciembre de 2014). [Online] http://www.ricoh.com/r_dc/photostyle/knowledge/basic/white/whats.html.

[2]. Cambios en el color inducido por la luz solar. (accedido en Diciembre 2014). [Online] http://www.apogeephoto.com/july2004/jaltengarten7_2004.shtml.

[3]. Temperatura de Color. (accedido en diciembre 2014). [En línea] http://www.lrc.rpi.edu/programs/nlpip/lightinganswers/lightsources/whatisCCT.asp .

[4]. Cómo convertir Temperatura de Color (K) a sus componentes RGB. (accedido en Diciembre 2014). [En línea] http://www.tannerhelland.com/4435/convert-temperature-rgb-algorithm-code/ .

[5]. Espacios de color. (accedido en febrero 2015). [En línea] https://www.lacie.com/download/whitepaper/wp_colormanagement_3_es.pdf

[6]. Espacios de color II. (accedido en diciembre 2014). [En línea] http://bibing.us.es/proyectos/abreproy/11875/fichero/Proyecto+Fin+de+Carrera%252F3.Espacios+de+color.pdf.

[7]. Primarias en el espacio CIE1931. (accedido en marzo 2015). [En línea] http://hyperphysics.phy-astr.gsu.edu/hbase/vision/cieprim.html.

[8]. Espacio de color CIE1931. (accedido en marzo 2015). [En línea] http://www.ppsloan.org/publications/XYZJCGT.pdf .

[9]. **Wannous Hazem [et al.]** Improving color correction across camera and illumination changes by contextual sample selection [Journal] // Journal of Electronic Imaging.- [s.l.]: International Society for Optics and Photonics, 2012.- 2: Vol. 21.- pp. 23015-1

[10]. **G Zapryanov D Ivanova, I Nikolova** Automatic White Balance Algorithms for Digital Still Cameras – a Comparative Study [Publicación periódica]// information technologies and control.– 2012

[11]. **Pascale Danny** Rgb coordinates of the macbeth color checker [Journal]// Rapport technique, The BabelColor Company.- 2006.- p. 64.

[12]. **Summanen Kasperi** Application for Subjective Auto White Balance Accuracy Measurement [Publicación periódica]. – 2013

[13]. **Chiu Li-Cheng and Fuh Chiou-Shann** Calibration-based auto white balance method for digital still camera [Journal] // Journal of Information Science and Engineering. - 2010. - 2 : Vol. 26. - pp. 713-723.

[14]. **Chen Cheng-Lun and Lin Shao-Hua** Intelligent color temperature estimation using fuzzy neural network with application to automatic white balance [Journal] // Expert Systems with Applications . - 2011. - 6 : Vol. 38. - pp. 7718-7728. - ISSN: 0957-4174 DOI: http://dx.doi.org/10.1016/j.eswa.2010.12.137.

[15]. El color es luz (accedido en diciembre 2014): http://www.proyectacolor.cl/teoria-de-los-colores/el-color-es-luz/

[16]. Formación de colores (accedido en diciembre 2014): http://www.fotonostra.com/grafico/coloresobjetos.htm

[17]. **Botero Valencia J.-S., Lopez Giraldo F.-E. and Vargas Bonilla J.-F.** Calibration method for Correlated Color Temperature (CCT) measurement using RGB color sensors [Conference] // Image, Signal Processing, and Artificial Vision (STSIVA), 2013 XVIII Symposium of.- 2013.- pp. 1-6.- DOI: 10.1109/STSIVA.2013.6644921.

[18]. **Smith, Joe**. Calculating Color Temperature and Illuminance using the TAOS TCS3414CS Digital Color Sensor. Designer's Notebook. Intelligent opto sensors, number 25, TAOS. Rev C. February 2009.

[19]. Hoja de características sensor TAOS TCS3414CS (accedido en marzo 2015): http://datasheet.octopart.com/TCS3414CS-TAOS-datasheet-17017938.pdf.

[20]. Hoja de características sensor TAOS TCS34725 (accedido en marzo 2015): http://www.adafruit.com/datasheets/TCS34725.pdf.

[21]. **Xiao, F., Farrell, J. E., DiCarlo, J. M., & Wandell, B. A.** Preferred color spaces for white balancing. In Electronic Imaging 2003 (pp. 342-350). International Society for Optics and Photonics.

[22]. Características espectrales del color y su evaluación en fuentes de iluminación (accedido en diciembre 2014): http://juliogm.wordpress.com/2014/08/13/caracteristicas-espectrales-del-color-y-su-evaluacion-en-fuentes-de-iluminacion/ .

[23]. Índice de Reproducción Cromática (CRI) (accedido en diciembre 2014): http://www.schreder.com/cls-es/Centro-Formacion/Esencial-Alumbrado/Pages/Colour-rendering.aspx.

[24]. Índice de Reproducción Cromática (CRI) (accedido en diciembre 2014): http://lediagroup.com/tecnologia-led/el-color-depende-del-led-con-que-se-ilumine/ .

[25]. CRI ejemplos (accedido en diciembre 2014): http://www.liquidleds.com.au/blog/common-cri-questions.

[26]. Emisión espectral de una fuente de luz (accedido en diciembre 2014): http://www.schreder.com/cls-es/Centro-Formacion/Esencial-Alumbrado/Pages/Spectral-distribution-of-a-light-source.aspx

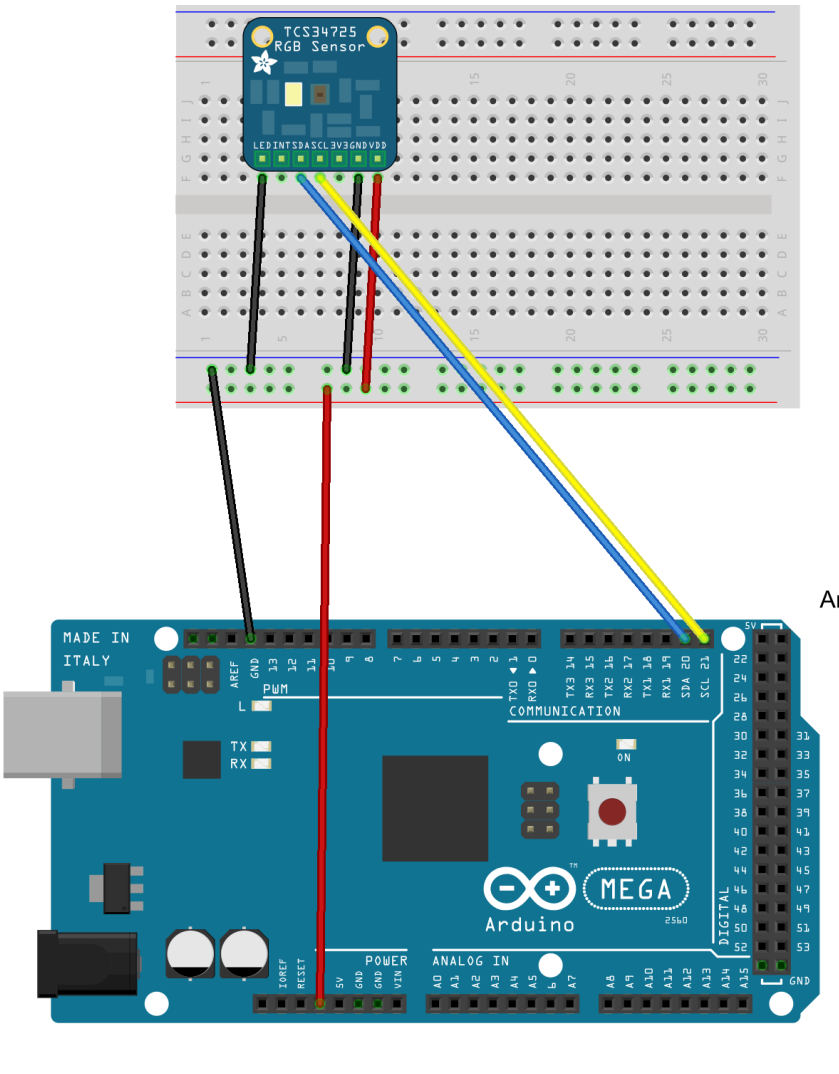
[27]. **James, G., Witten, D., Hastie, T., & Tibshirani**, R. (2013). An introduction to statistical learning (p. 6). New York: Springer.

[28]. **VERGÉS LLAHÍ, Jaume, et al.** Color constancy and image segmentation techniques for applications to mobile robotics. 2005.

[29]. TCS34725, características de producto y precio en la tienda de electrónica online CETRONIC: http://www.cetronic.es/sqlcommerce/disenos/plantilla1/seccion/producto/DetalleProducto.jsp?idIdioma=&idTienda=93&codProducto=888304046&cPath=1343 (accedido en Junio 2015).

7. Anexos

7.1. Anexo 1: Esquema de conexiones

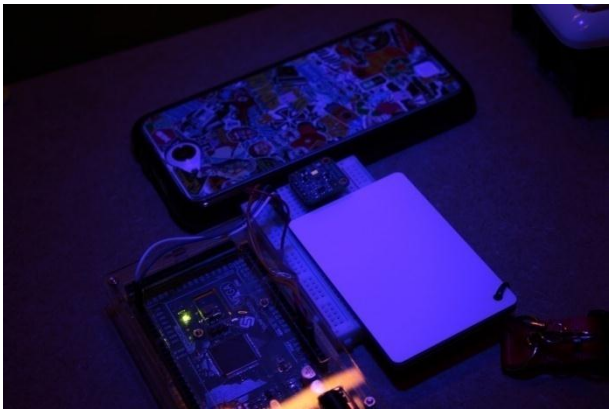
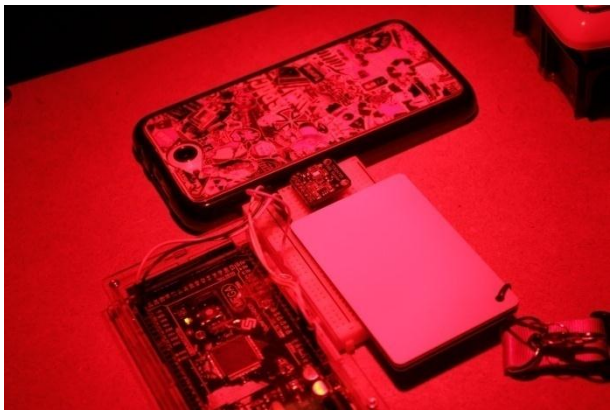
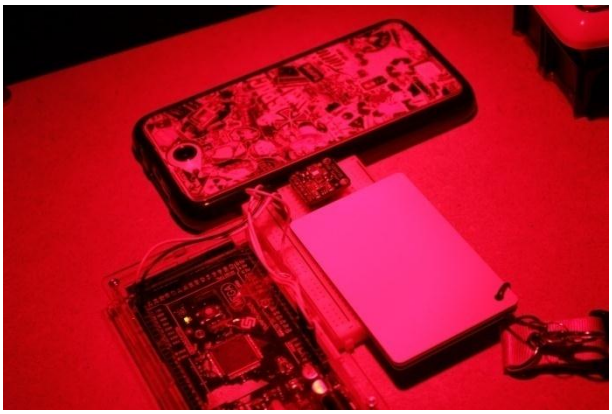


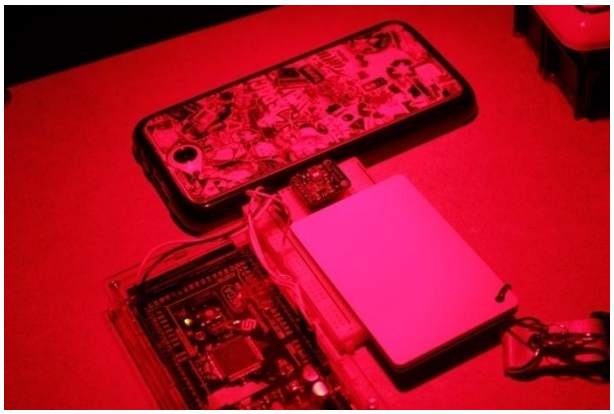
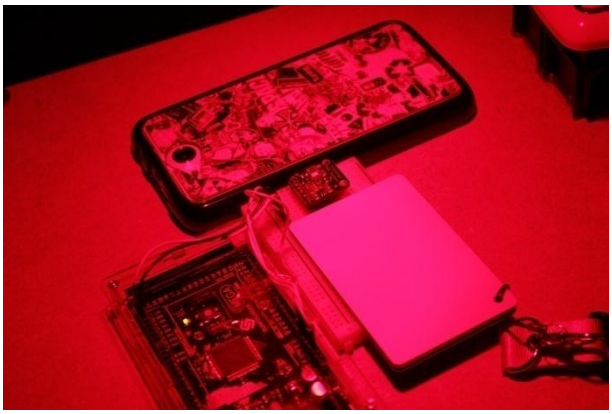
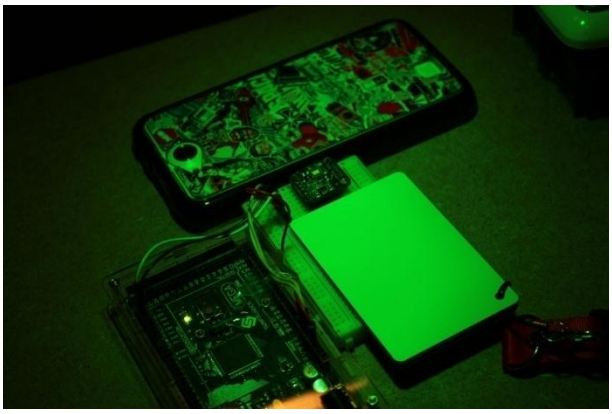
7.2. Anexo 2: Colección de fotografías de entrenamiento

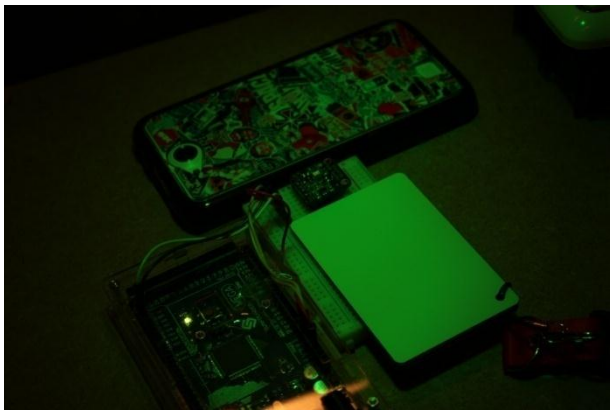
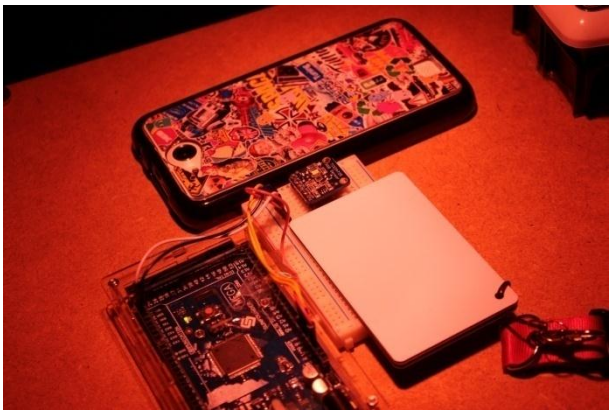
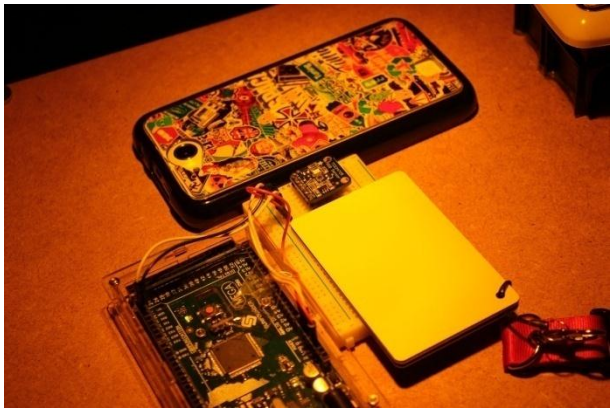
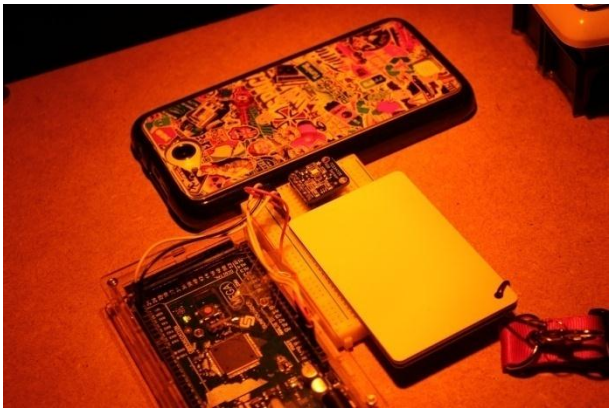
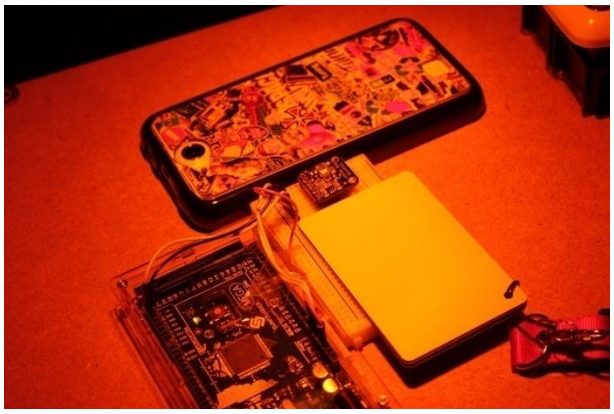
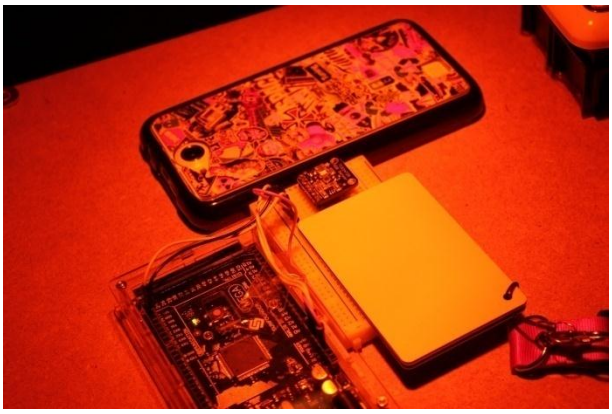
DISTANCIA VARIABLE	
	
Im1	Im2
	
Im3	Im4
	
Im5	Im6

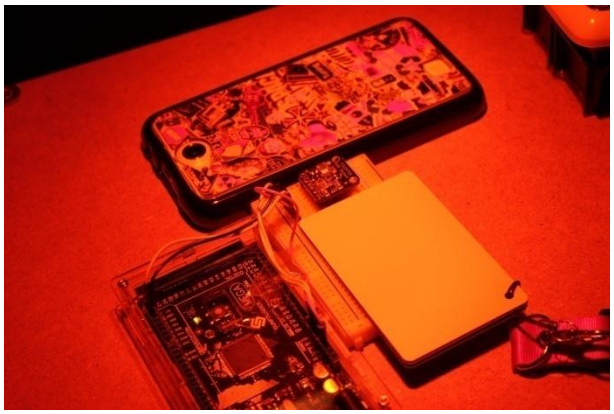
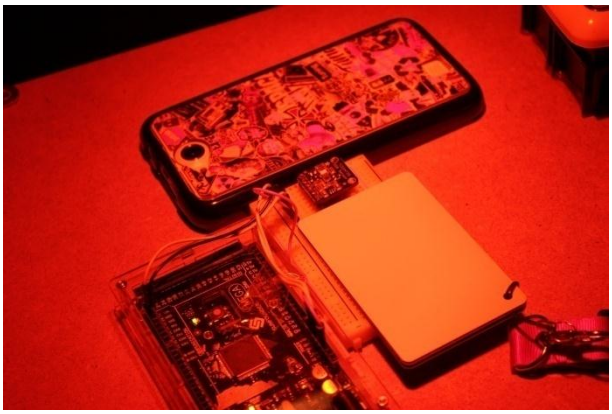
	
Im7	Im8
	
Im9	Im10
	
Im11	Im12

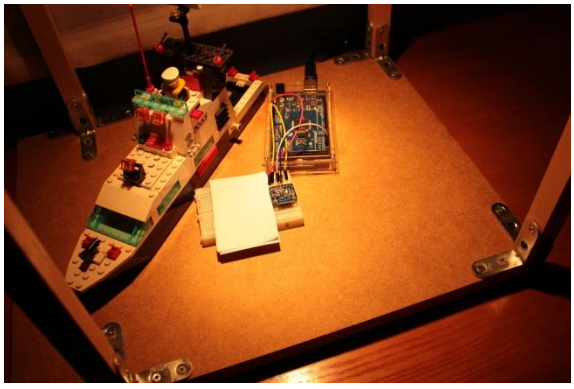
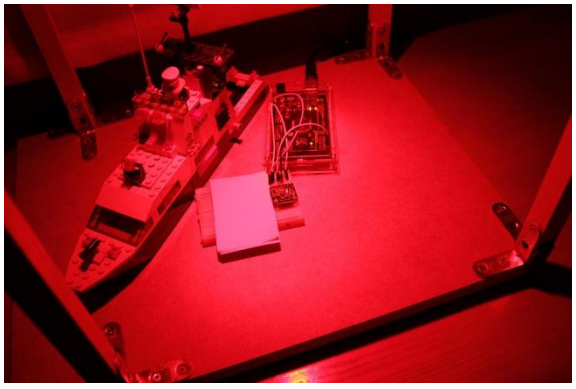
	
Im13	Im14
	
Im15	Im16
	
Im17	

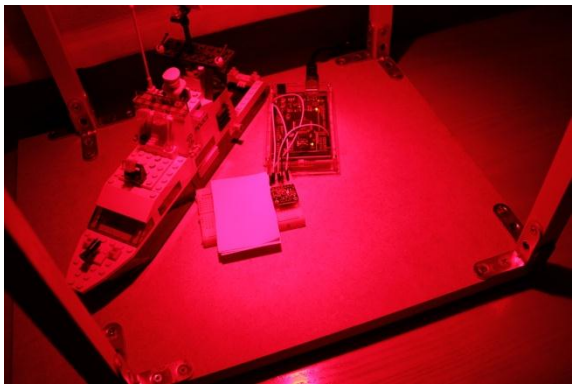
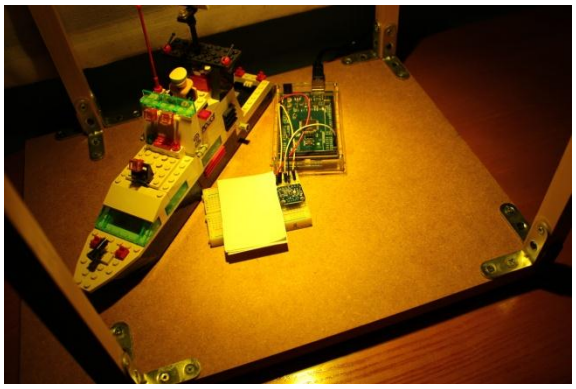
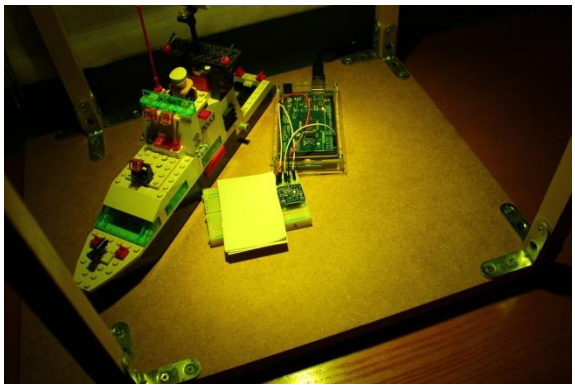
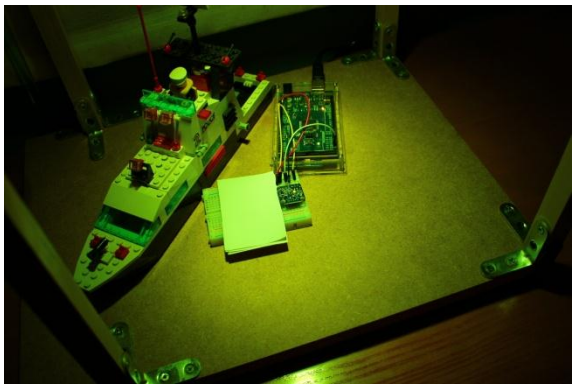
DISTANCIA FIJA, FILTROS VARIABLES	
	
Im18	Im19
	
Im20	Im21
	
Im22	Im23

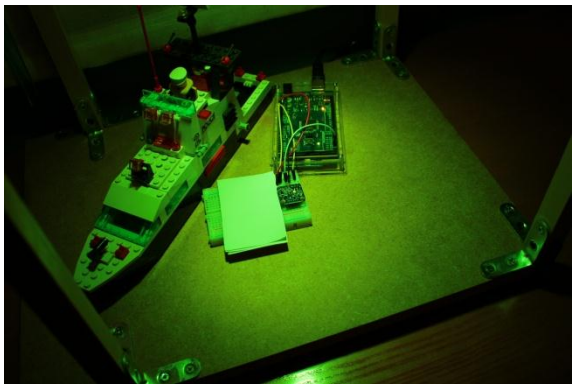
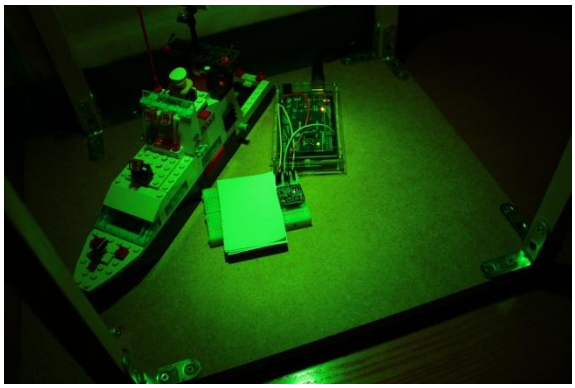
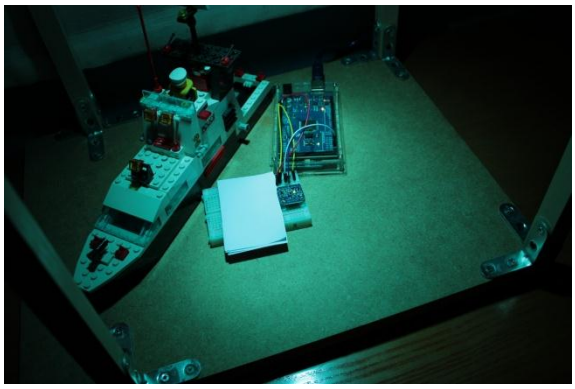
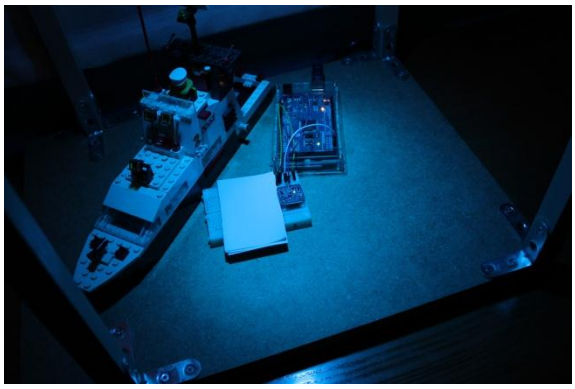
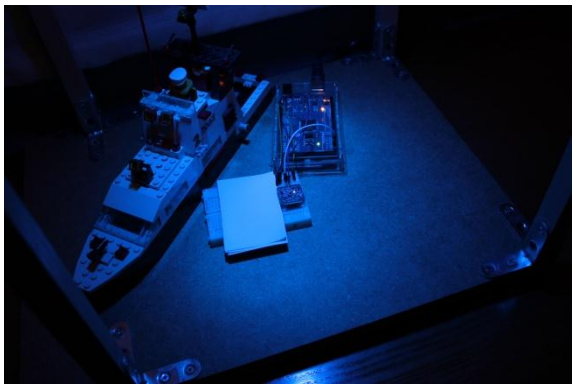
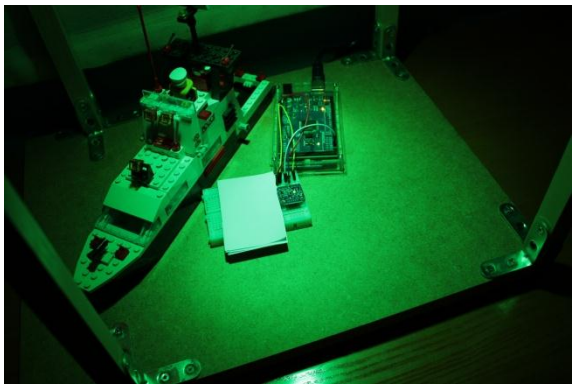
	
Im24	Im25
	
Im26	Im27
	
Im28	Im29

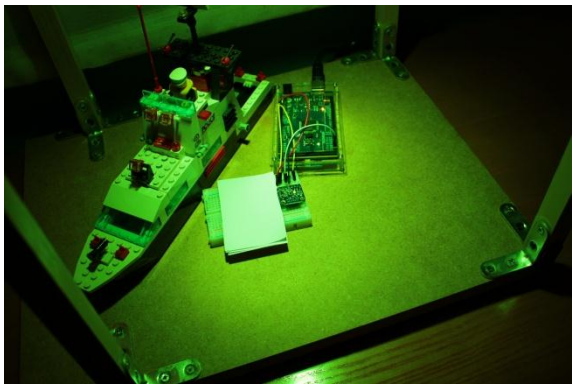
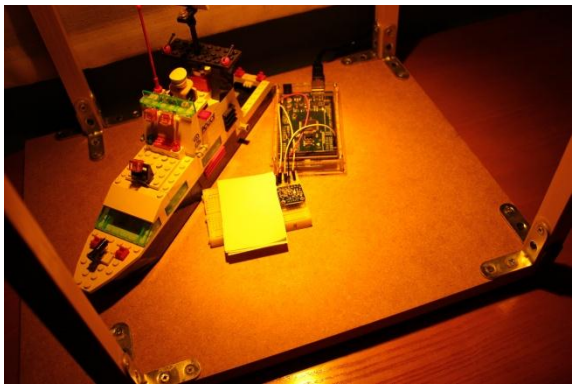
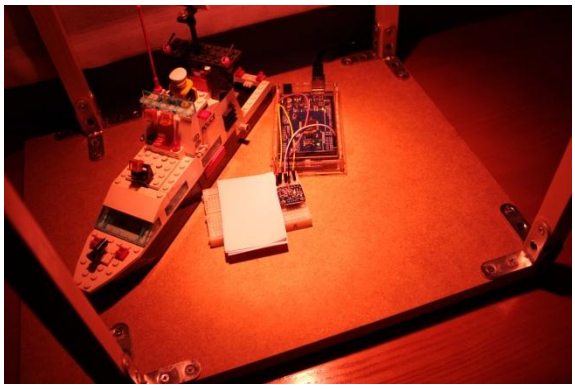
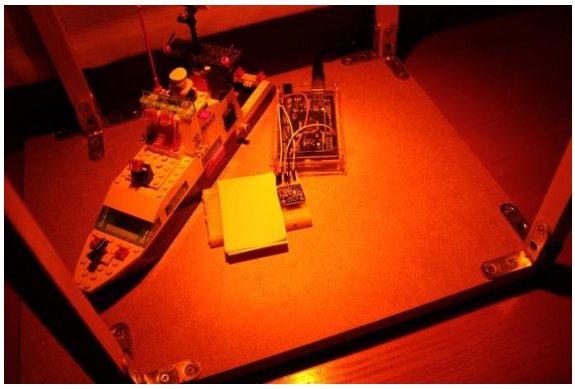
	
Im30	Im31
	
Im32	Im33
	
Im34	Im35

	
Im36	Im37
	
Im38	

OTROS	
	
Im39	Im40

	
Im41	Im42
	
Im43	Im44
	
Im45	Im46

	
Im47	Im48
	
Im49	Im50
	
Im51	Im52

	
Im53	Im54
	
Im55	Im56
	
Im57	Im58

	
Im59	Im60

7.3. Anexo 3: valores obtenidos de las imágenes de entrenamiento

Nombre	Descripción	R	G	B	Clear	Temp	Lux	Distancia
Im1	sin filtro lejos	6505	4853	4420	8939	4305	2278	Variable
Im2	sin filtro medio	10237	7853	7209	13790	4688	3797	Variable
Im3	sin filtro cerca	13568	10916	10010	18299	4984	5477	Variable
Im4	filtro rojo lejos	5499	3524	3647	5771	4306	1073	Variable
Im5	filtro rojo medio	8146	5360	5516	8358	4815	1751	Variable
Im6	filtro rojo cerca	11547	8289	8452	11845	5734	3166	Variable
Im7	filtro verde lejos	4616	3862	3528	5695	5310	2018	Variable
Im8	filtro verde medio	8043	6819	6227	9720	5525	3600	Variable
Im9	filtro verde cerca	10834	9411	8699	12879	5755	5025	Variable
Im10	rojo cerca 2 filtros	9169	6710	6839	9161	6192	2565	Variable
Im11	rojo cerca 3 filtros	6334	5036	5191	6461	6398	1943	Variable
Im12	amarillo lejos	6787	4805	4152	8641	3465	2357	Variable
Im13	amarillo medio	8107	5997	5185	10229	3773	3012	Variable
Im14	naranja lejos	7154	4768	4433	8334	3520	1935	Variable
Im15	naranja cerca	9640	6803	6416	11312	4025	2997	Variable
Im16	azul lejos	5166	4777	4896	6009	9091	2279	Variable
Im17	azul cerca	7611	7151	7287	8707	8939	3505	Variable
Im18	azul 1 filtro	27101	24959	25025	33742	8896	11818	Fija
Im19	azul 2 filtros	20424	17915	18795	21746	9684	7848	Fija
Im20	azul 3 filtros	16302	14324	14835	16993	9060	6401	Fija
Im21	azul 4 filtros	13570	11943	12238	13885	8786	5420	Fija

Im22	rojo 1 filtro	45073	23934	25331	47805	1792	4612	Fija
Im23	rojo 2 filtros	36551	20002	21251	37971	2210	4191	Fija
Im24	rojo 3 filtros	29625	16695	17663	30514	2695	3838	Fija
Im25	rojo 4 filtros	23772	13809	14593	24562	4149	3420	Fija
Im26	verde 1 filtro	38660	30005	26513	49620	4358	15349	Fija
Im27	verde 2 filtros	24646	21239	20417	27882	6366	10577	Fija
Im28	verde 3 filtros	18802	16196	16124	20124	7194	7694	Fija
Im29	verde 4 filtros	16010	13613	13750	16547	7558	6271	Fija
Im30	verde 5 filtros	13442	11431	11486	13661	7759	5126	Fija
Im31	rosa 1 filtro	51240	28965	28962	62470	2173	7722	Fija
Im32	amarillo 1 filtro	54890	34462	28995	65535	2538	15285	Fija
Im33	naranja 1 filtro	52874	30011	27598	62834	2114	9855	Fija
Im34	naranja 2 filtros	41744	22964	22031	47612	1980	6554	Fija
Im35	naranja 3 filtros	34007	18565	18354	37817	1967	4843	Fija
Im36	naranja 4 filtros	27453	15055	15200	29415	2048	3748	Fija
Im37	naranja 5 filtros	22879	12692	12966	24642	2220	3105	Fija
Im38	azul + verde	24087	21606	21335	26683	7505	10645	Fija
Im39	Sin filtro	53872	36038	32526	65535	3336	*	
Im40	Rojo 1 filtro	42235	22705	23955	44439	1940	*	
Im41	Rojo 2 filtros	38031	20732	21994	39683	2126	*	
Im42	Rojo 3 filtros	34919	19355	20534	36293	2364	*	
Im43	Rojo 4 filtros	31042	17540	18586	32172	2661	*	
Im44	Verde 1 filtro	36717	28730	25506	47094	4447	*	
Im45	Verde 2 filtros	28193	23800	22198	33698	5690	*	
Im46	Verde 3 filtros	23822	20519	19739	26905	6455	*	
Im47	Verde 4 filtros	21142	18213	17886	22995	6948	*	
Im48	Verde 5 filtros	18805	16128	16057	19915	7281	*	
Im49	Azul 1 filtro	26776	24636	25220	33460	8990	*	
Im50	Azul 2 filtros	22576	19981	21196	25164	10281	*	
Im51	Azul 3 filtros	20230	17720	18695	21678	9842	*	
Im52	Azul + verde 2 filtros	23850	21319	21188	26497	7646	*	
Im53	Azul + amarillo 2 filtros	24451	21638	21045	27311	6957	*	
Im54	Amarillo 1 filtro	52223	33009	27879	65535	2618	*	
Im55	Rosa 1 filtro	50069	28351	28317	60351	2324	*	
Im56	Naranja 1 filtro	48827	27854	25740	58529	2159	*	

Im57	Naranja 2 filtros	39810	22142	21289	45562	2053	*	
Im58	Naranja 3 filtros	32601	18064	17885	36246	2085	*	
Im59	Naranja 4 filtros	27561	15339	15487	30063	2184	*	
Im60	Naranja 5 filtros	22722	12812	13103	24432	2386	*	

7.4. Anexo 4: Valores obtenidos a través de la cámara para las imágenes de entrenamiento

Nombre	Rtest	Gtest	Btest
Im1	255	240	176
Im2	255	241	177
Im3	255	242	185
Im4	255	102	162
Im5	255	111	163
Im6	255	121	158
Im7	255	253	65
Im8	255	254	75
Im9	255	255	88
Im10	151	23	59
Im11	148	20	61
Im12	144	121	11
Im13	158	135	16
Im14	235	79	8
Im15	239	83	7
Im16	140	225	237
Im17	144	226	238
Im18	37	205	220
Im19	9	129	255
Im20	71	89	255
Im21	101	60	255
Im22	255	79	130
Im23	255	56	141
Im24	255	47	145
Im25	255	38	132
Im26	255	217	15

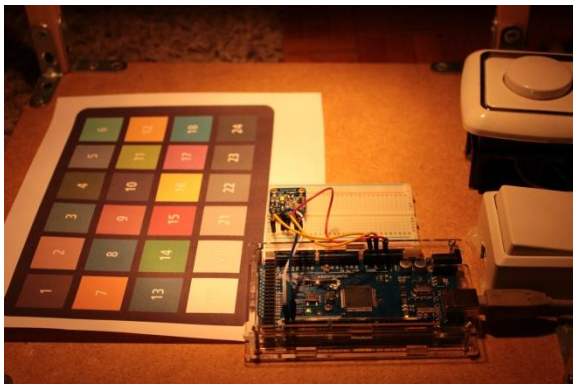
Im27	159	236	43
Im28	33	231	45
Im29	3	208	29
Im30	12	129	17
Im31	255	167	135
Im32	255	207	0
Im33	255	187	11
Im34	255	165	4
Im35	255	154	16
Im36	255	135	31
Im37	255	123	50
Im38	46	233	153
Im39	247	161	66
Im40	255	36	54
Im41	250	23	53
Im42	251	22	60
Im43	251	22	60
Im44	217	171	3
Im45	161	172	7
Im46	100	169	9
Im47	48	168	11
Im48	17	154	14
Im49	11	157	144
Im50	1	120	199
Im51	3	86	201
Im52	2	189	80
Im53	64	197	27
Im54	254	152	0
Im55	255	112	65
Im56	255	121	3
Im57	255	97	3
Im58	255	81	7
Im59	255	75	6
Im60	255	57	17

7.5. Anexo 5: Datos medidos por el sensor para última prueba

Tipo	R	G	B	C	CCT
Final1	54565	36034	32488	65535	3242
Final2	33101	18453	19452	34321	2346
Final3	22303	19236	18356	25092	6386
Final4	19337	16887	17800	20591	9749

7.6. Anexo 6: Imágenes para última prueba y colores extraídos de la misma

FINALES

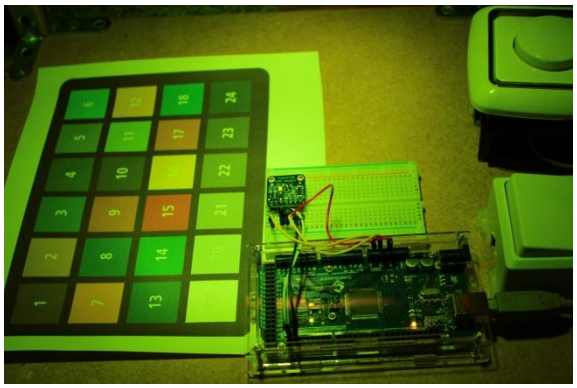


Final1

Final1

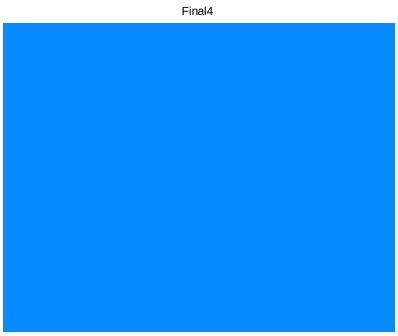
Final2

Final2



Final3

Final4



7.7. Anexo 7: Valores obtenidos por la cámara para las imágenes de test finales

Tipo	R	G	B
Final1	255	228	148
Final2	255	72	180
Final3	172	246	49
Final4	7	139	255

7.8. Anexo 8: Arduino code

```
/*
  Lectura del sensor RGB TCS34725 y comunicación serial de esos datos.
  Placa basada en Arduino Mega 2560 rev3.
*/
```

```
#include <Wire.h>
#include "Adafruit_TCS34725.h"

/* Se inicializa con unos valores específicos de tiempo y ganancias */
Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_700MS, TCS34725_GAIN_1X);

//=========================================================================
//  SETUP
//=========================================================================
void setup(void) {
 Serial.begin(9600);

 if (tcs.begin()) {
  Serial.println("Found sensor");
 } else {
  Serial.println("No TCS34725 found ... check your connections");
  while (1);
 }

}

//=========================================================================
//  LOOP
//=========================================================================
void loop(void) {
 uint16_t r, g, b, c, colorTemp, lux;
 int rb,gb,bb;
```

```
tcs.getRawData(&r, &g, &b, &c);
colorTemp = tcs.calculateColorTemperature(r, g, b);
lux = tcs.calculateLux(r, g, b);

//  Comunicación serie del valor capturado por el sensor para el CANAL ROJO
Serial.print(r, DEC); Serial.print("\n");
//  Comunicación serie del valor capturado por el sensor para el CANAL VERDE
Serial.print(g, DEC); Serial.print("\n");
//  Comunicación serie del valor capturado por el sensor para el CANAL AZUL
Serial.print(b, DEC); Serial.print("\n");
// Comunicación serie del valor capturado por el sensor para LA ILUMINACIÓN
Serial.print(c,DEC); Serial.print("\n");
//  Comunicación serie del valor capturado por el sensor para LA TEMPERATURA DE COLOR
Serial.print(colorTemp, DEC); Serial.print("\n");
// //  Comunicación serie del valor capturado por el sensor para LUXES
Serial.print("Lux: "); Serial.print(lux, DEC); Serial.print("\n");
// conversion de datos en 8 bit
rb = int(255*double(r)/65535);
gb = int(255*double(g)/65535);
bb = int(255*double(b)/65535);
// Envío a través de serie de los datos en 8 bits
Serial.print("r8: ");Serial.print(rb,DEC);Serial.print("\n");
Serial.print("g8: ");Serial.print(gb,DEC);Serial.print("\n");
Serial.print("b8: ");Serial.print(bb,DEC);Serial.print("\n");

delay(1000); // Se hace una espera de 1000ms para el siguiente ciclo
}
```

7.9. Anexo 9: Programa principal de ejecución (script MATLAB)

```
% Programa principal
close all;
clear;
clc;
%% Datos para su uso
Cfinal1 = [54565    36034    32488    65535    3242];
Cfinal2 = [33101    18453    19452    34321    2346];
Cfinal3 = [22303    19236    18356    25092    6386];
Cfinal4 = [19337    16887    17800    20591    9749];

Im1final1 = 'Final1.JPG';
Im1final2 = 'Final2.JPG';
Im1final3 = 'Final3.JPG';
Im1final4 = 'Final4.JPG';

% Neurona
net1 = 'NN20_br.mat';
net2 = 'NN15_lm.mat';
net3 = 'NN30_lm.mat';
net = load(net2);
net = net.net;

bits = 8; % Desarrollo en RGB 8 bits

%% Ejecución principal
CAR = Obten_caracteristicas(1,Cfinal4);
Color = Adecua_valoresRGB(net,CAR);
Color2 = uint8(Color);
Im = Obten_imagen(1, Im1final4);
Im_adj = WB_image(Im,Color(1),Color(2),Color(3),bits);
Muestra_resultados(Im,Im_adj,Color);
```

7.10. Anexo 10: Función de obtención de características de la luz incidente (script MATLAB)

```
function CAR = Obten_caracteristicas(param,carac)

% Si param == 0, las características se tomarán del dispositivo externo
% Si param == 1, las características se replican de los parámetros
% indicados

if param == 0
   CAR = Lee_serie();
end
if param == 1
```

```
    CAR = carac;
end
```

7.11. Anexo 11: Lectura serie (script MATLAB)

```
% Esta función permite leer los datos serie que son enviados desde la
% tarjeta Arduino Mega2560.
function [vR,vG,vB,vC,CCT] = Lee_serie()

% Se crea el objeto puerto serie correspondiente
s = serial('COM6');
% Se abre tal objeto
fopen(s);
% Se lee la primera línea que muestra "44" que no se necesita
line1=fgets(s);
% Se lee la segunda línea que se corresponde con "Found sensor"
line2=fgets(s);

% Las dos primeras líneas leidas no representan datos útilies, por lo que
% van a ser desechadas

% Se lee la tercera línea que es en la que ya se escribe el primer valor de
% los leídos por el sensor, el correspondiente al Rojo
vR=fgets(s);
vR=uint16(str2num(vR)); % Se transforma a un entero 16 bits

% Se lee la cuarta línea que es el valor del verde
vG=fgets(s);
vG=uint16(str2num(vG)); % Se transforma a un entero 16 bits

% Se lee la quinta línea que es el valor del azul
vB=fgets(s);
vB=uint16(str2num(vB)); % Se transforma a un entero 16 bits

% Se lee el canal claro
vC = fgets(s);
vC=uint16(str2num(vC)); % Se transforma a un entero 16 bits

% Se lee el CCT calculado en el Arduino
CCT = fgets(s);
CCT=uint16(str2num(CCT)); % Se transforma a un entero 16 bits

% Se cierra el puerto para que otros puedan hacer uso
fclose(s);
```

7.12. Anexo 12: Adecuación de valores RGB (script MATLAB)

```
% Función que dada una red neuronal y un vector de datos (vr,vg,vb,vc,CCT)
% predice los valores a reales de RGB a través de tal red neuronal
function Color = Adecua_valoresRGB(net,data)
aux = sim(net,data');


if (aux(1)>255)
    aux(1)= 255;
end
if (aux(1)<0)
    aux(1)=0;
end

if (aux(2)>255)
    aux(2)= 255;
end
if (aux(2)<0)
    aux(2)=0;
end

if (aux(3)>255)
    aux(3)= 255;
end
if (aux(3)<0)
    aux(3)=0;
end

Color=aux;
```

7.13. Anexo 13: Obtención de imagen (script MATLAB)

```
% Función que permita la obtención de una imagen y la devuelva

function Im = Obten_imagen(param, nombre_im)
% Si param == 0, la imagen tiene que se capturada a través de la webcam y
% por tanto se utizará la función Take_snap()
% Si param == 1, la imagen será la imagen cargada con el nombre de la misma

if param == 0
    Im = Take_snap();
end
if param == 1
    Im = imread(nombre_im);
end
```

7.14. Anexo 14: Función de captura de imágenes (script MATLAB)

```
function photo = Take_snap()
% Función que devuelve una imagen tomada a través de una webcam conectada
% al PC.
% Selección del dispositivo de video
% imaqhwinfo('winvideo',1).SupportedFormats
vidobj = videoinput('winvideo', 2,  'RGB24_1280x960');
% Se toma la imagen
photo = getsnapshot(vidobj);
```

7.15. Anexo 15: Balance de imagen (script MATLAB)

```
% Función que permite realizar el Balance de Blancos sobre la imagen que se
% le envíe siguiendo el método Retinex.
% Las variables de entrada serán:
% -Im: imagen a ser corregida
% -vr: valor del canal rojo
% -vg: valor del canal verde
% -vb: valor del canal azul
% -bits: resolución en bits en la que vienen expresados los canales de la
%  imagen y los valores de vr, vg y vb para corrección
function Im_adj = WB_image(Im,vr,vg,vb,bits)

% Se hace casting de los datos a formato double
Im=double(Im);
vr=double(vr);
vg=double(vg);
vb=double(vb);

% Se calculan el número de filas y columnas, sabiendo que la tercera
% dimensión es 3 por debida a los 3 canales de color RGB
a = size(Im(:,:,1),1);
b = size(Im(:,:,2),2);
Limit = (2^bits)-1;

% Se obtienen los coeficientes de ajuste. Este coeficiente de ajuste hará
% la extrapolación de los datos de forma que la escala vuelva a 0-255,
% teniendo en cuenta que el sensor se entiende como una superfice blanca
% perfecta que es capaz, en este caso, de absorber todas las longitudes de
% onda
coefR = 255/vr;
coefG = 255/vg;
coefB = 255/vb;

% Se aplican los coeficientes en cada canal para terminar la corrección de
% la misma
Im_adj(:,:,1) = coefR.*Im(:,:,1);
Im_adj(:,:,2) = coefG.*Im(:,:,2);
Im_adj(:,:,3) = coefB.*Im(:,:,3);
```

```
Im_adj = uint8(Im_adj);
```

7.16. Anexo 16: Mostrar los resultados (script MATLAB)

```
% Función que permite mostrar los resultados

function  Muestra_resultados(Im1,Im2,Color)

figure;
% Se representa la imagen original
subplot(1,2,1);
imagesc(Im1);
title('ORIGINAL');

% Se representa la imagen ajustada
subplot(1,2,2);
imagesc(Im2);
title('AJUSTADA');

% Se representa un cuadro con el color capturado
figure;
a = ones(50,50,3);
a(:,:,1) = a(:,:,1).*Color(1);
a(:,:,2) = a(:,:,2).*Color(2);
a(:,:,3) = a(:,:,3).*Color(3);
a = uint8(a);
imagesc(a);
title('COLOR CAPTURADO');
```

7.17. Anexo 17: Script de MATLAB auxiliar para cálculos de regresiones

```
%% SCRIPT DE CARACTERIZACIÓN
% Este script de Matlab permitirá obtener los métodos de regresión
% adecuados para la caracterización y calibración del sensor utilizado
clear;
clc
close all;


%% Entrada del sensor

% Los datos que se presentan a continuación han sido recogidos a través de
% los test realizados al sensor
```

```
Datos_sens = [6505  4853    4420    8939    4305;
10237   7853    7209    13790   4688;
13568   10916   10010   18299   4984;
5499    3524    3647    5771    4306;
8146    5360    5516    8358    4815;
11547   8289    8452    11845   5734;
4616    3862    3528    5695    5310;
8043    6819    6227    9720    5525;
10834   9411    8699    12879   5755;
9169    6710    6839    9161    6192;
6334    5036    5191    6461    6398;
6787    4805    4152    8641    3465;
8107    5997    5185    10229   3773;
7154    4768    4433    8334    3520;
9640    6803    6416    11312   4025;
5166    4777    4896    6009    9091;
7611    7151    7287    8707    8939;
27101   24959   25025   33742   8896;
20424   17915   18795   21746   9684;
16302   14324   14835   16993   9060;
13570   11943   12238   13885   8786;
45073   23934   25331   47805   1792;
36551   20002   21251   37971   2210;
29625   16695   17663   30514   2695;
23772   13809   14593   24562   4149;
38660   30005   26513   49620   4358;
24646   21239   20417   27882   6366;
18802   16196   16124   20124   7194;
16010   13613   13750   16547   7558;
13442   11431   11486   13661   7759;
51240   28965   28962   62470   2173;
54890   34462   28995   65535   2538;
52874   30011   27598   62834   2114;
41744   22964   22031   47612   1980;
34007   18565   18354   37817   1967;
27453   15055   15200   29415   2048;
22879   12692   12966   24642   2220;
24087   21606   21335   26683   7505;
53872   36038   32526   65535   3336;
42235   22705   23955   44439   1940;
38031   20732   21994   39683   2126;
34919   19355   20534   36293   2364;
31042   17540   18586   32172   2661;
36717   28730   25506   47094   4447;
28193   23800   22198   33698   5690;
23822   20519   19739   26905   6455;
21142   18213   17886   22995   6948;
18805   16128   16057   19915   7281;
26776   24636   25220   33460   8990;
22576   19981   21196   25164   10281;
20230   17720   18695   21678   9842;
23850   21319   21188   26497   7646;
24451   21638   21045   27311   6957;
52223   33009   27879   65535   2618;
50069   28351   28317   60351   2324;
```

```
48827   27854   25740   58529   2159;
39810   22142   21289   45562   2053;
32601   18064   17885   36246   2085;
27561   15339   15487   30063   2184;
22722   12812   13103   24432   2386;
];

Datos_sen = Datos_sens;

% Obtenidos sobre la cámara
Color =  [255   240 176;
255 241 177;
255 242 185;
255 102 162;
255 111 163;
255 121 158;
255 253 65;
255 254 75;
255 255 88;
151 23  59;
148 20  61;
144 121 11;
158 135 16;
235 79  8;
239 83  7;
140 225 237;
144 226 238;
37  205 220;
9   129 255;
71  89  255;
101 60  255;
255 79  130;
255 56  141;
255 47  145;
255 38  132;
255 217 15;
159 236 43;
33  231 45;
3   208 29;
12  129 17;
255 167 135;
255 207 0;
255 187 11;
255 165 4;
255 154 16;
255 135 31;
255 123 50;
46  233 153;
247 161 66;
255 36  54;
250 23  53;
251 22  60;
251 22  60;
217 171 3;
161 172 7;
100 169 9;
```

```
48  168 11;
17  154 14;
11  157 144;
1   120 199;
3   86  201;
2   189 80;
64  197 27;
254 152 0;
255 112 65;
255 121 3;
255 97  3;
255 81  7;
255 75  6;
255 57  17;
];



%% Reparto aleatorio
% Se realiza un reparto aleatorio de los 60 datos existentes para que
% deshacer el orden con el que fueron realizados los test y de esta forma
% conseguir un grupo más heterogéneo de cara al reparto en grupos de
% entrenamiento, validación y test

ale = [16
    38
55
25
18
45
24
50
 3
58
20
51
 4
49
32
37
 1
33
54
60
39
23
31
56
 9
30
41
 7
22
34
10
26
```

```
19
42
47
13
28
44
14
12
 6
46
 8
27
11
35
17
52
15
21
 5
40
57
59
48
43
 2
36
53
29
];

% Reparto de los datos en grupos
n_datos = size(ale,1);
datxgrup = floor(n_datos/4);

Datos_senb = Datos_sen(ale(:),:); % Se reordenan los datos con la distribución
aleatoria
Colorb = Color(ale(:),:);

% Se dividen en grupos para posterior análisis
Datos_sen1 = [Datos_senb(1:datxgrup,:)];
Datos_sen2 = [Datos_senb((datxgrup+1):(2*datxgrup),:)];
Datos_sen3 = [Datos_senb((2*datxgrup+1):(3*datxgrup),:)];
Datos_sen4 = [Datos_senb((3*datxgrup+1):n_datos,:)];

Color1 = [Colorb(1:datxgrup,:)];
Color2 = [Colorb((datxgrup+1):(2*datxgrup),:)];
Color3 = [Colorb((2*datxgrup+1):(3*datxgrup),:)];
Color4 = [Colorb((3*datxgrup+1):n_datos,:)];

%% Multi variable linear regresion & MARS
% =======================================

% En este apartado lo que se hará será probar las diferentes opciones que
% hay de combinación de variables a ser utilizadas que serán 25
% posibilidades distintas, haciendo un proceso de validación cruzada en
```

```
% cada uno de los casos, con el error cuadrático medio calculado sobre uno
% de los conjuntos.

% Parte lineal
% Todas las variables para rojo
DATA1 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 1, 1,1)
DATA2 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 1, 1,1)
DATA3 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 1, 1,1)
DATA4 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 1, 1,1)
% Todas las variables para verde
DATA5 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 1, 1,2)
DATA6 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 1, 1,2)
DATA7 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 1, 1,2)
DATA8 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 1, 1,2)
% Todas las variables para azul
DATA9 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 1, 1,3)
DATA10 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 1, 1,3)
DATA11 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 1, 1,3)
DATA12 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 1, 1,3)
% Rojo para rojo
DATA13 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 0, 0, 0, 0,1)
DATA14 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 0, 0, 0, 0,1)
DATA15 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 0, 0, 0, 0,1)
DATA16 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 0, 0, 0, 0,1)
% Verde para verde
DATA17 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 1, 0, 0, 0,2)
DATA18 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 1, 0, 0, 0,2)
DATA19 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 1, 0, 0, 0,2)
DATA20 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 1, 0, 0, 0,2)
% Azul para azul
DATA21 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 0, 1, 0, 0,3)
DATA22 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 0, 1, 0, 0,3)
DATA23 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 0, 1, 0, 0,3)
```

```
DATA24 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 0, 1, 0, 0,3)
% Rojo verde azul para rojo
DATA25 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 0, 0,1)
DATA26 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 0, 0,1)
DATA27 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 0, 0,1)
DATA28 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 0, 0,1)
% Rojo verde azul para verde
DATA29 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 0, 0,2)
DATA30 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 0, 0,2)
DATA31 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 0, 0,2)
DATA32 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 0, 0,2)
% Rojo verde azul para azul
DATA33 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 0, 0,3)
DATA34 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 0, 0,3)
DATA35 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 0, 0,3)
DATA36 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 0, 0,3)
% Rojo verde azul clear para rojo
DATA37 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 1, 0,1)
DATA38 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 1, 0,1)
DATA39 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 1, 0,1)
DATA40 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 1, 0,1)
% Rojo verde azul claro para verde
DATA41 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 1, 0,2)
DATA42 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 1, 0,2)
DATA43 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 1, 0,2)
DATA44 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 1, 0,2)
% Rojo verde azul clear para azul
DATA45 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 1, 1, 1, 0,3)
DATA46 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 1, 1, 1, 0,3)
DATA47 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 1, 1, 1, 0,3)
DATA48 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 1, 1, 1, 0,3)
% Rojo clear y cct para rojo
```

```
DATA49 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 1, 0, 0, 1, 1,1)
DATA50 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 1, 0, 0, 1, 1,1)
DATA51 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 1, 0, 0, 1, 1,1)
DATA52 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 1, 0, 0, 1, 1,1)
% Verde clear y cct para verde
DATA53 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 1, 0, 1, 1,2)
DATA54 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 1, 0, 1, 1,2)
DATA55 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 1, 0, 1, 1,2)
DATA56 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 1, 0, 1, 1,2)
% Azul clear y cct para azul
DATA57 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 0, 1, 1, 1,3)
DATA58 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 0, 1, 1, 1,3)
DATA59 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 0, 1, 1, 1,3)
DATA60 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 0, 1, 1, 1,3)
% CCT para rojo
DATA61 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 0, 0, 0, 1,1)
DATA62 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 0, 0, 0, 1,1)
DATA63 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 0, 0, 0, 1,1)
DATA64 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 0, 0, 0, 1,1)
% CCT para verde
DATA65 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 0, 0, 0, 1,2)
DATA66 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 0, 0, 0, 1,2)
DATA67 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 0, 0, 0, 1,2)
DATA68 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 0, 0, 0, 1,2)
% CCT para azul
DATA69 = analiza(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1, Color2,
Color3, Color4, 0, 0, 0, 0, 1,3)
DATA70 = analiza(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1, Color2,
Color4, Color3, 0, 0, 0, 0, 1,3)
DATA71 = analiza(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1, Color3,
Color4, Color2, 0, 0, 0, 0, 1,3)
DATA72 = analiza(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2, Color3,
Color4, Color1, 0, 0, 0, 0, 1,3)

% Parte MARS
% Todas las variables para el rojo
```

```
DATA1B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 1, 1,1)
DATA2B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 1, 1,1)
DATA3B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 1, 1,1)
DATA4B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 1, 1,1)
% Todas las variables para verde
DATA5B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 1, 1,2)
DATA6B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 1, 1,2)
DATA7B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 1, 1,2)
DATA8B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 1, 1,2)
% Todas las variables para azul
DATA9B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 1, 1,3)
DATA10B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 1, 1,3)
DATA11B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 1, 1,3)
DATA12B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 1, 1,3)
% Rojo para rojo
DATA13B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 0, 0, 0, 0,1)
DATA14B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 0, 0, 0, 0,1)
DATA15B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 0, 0, 0, 0,1)
DATA16B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 0, 0, 0, 0,1)
% Verde para verde
DATA17B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 1, 0, 0, 0,2)
DATA18B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 1, 0, 0, 0,2)
DATA19B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 1, 0, 0, 0,2)
DATA20B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 1, 0, 0, 0,2)
% Azul para azul
DATA21B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 0, 1, 0, 0,3)
DATA22B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 0, 1, 0, 0,3)
DATA23B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 0, 1, 0, 0,3)
DATA24B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 0, 1, 0, 0,3)
% Rojo verde azul para rojo
DATA25B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 0, 0,1)
```

```
DATA26B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 0, 0,1)
DATA27B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 0, 0,1)
DATA28B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 0, 0,1)
% Rojo verde azul para verde
DATA29B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 0, 0,2)
DATA30B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 0, 0,2)
DATA31B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 0, 0,2)
DATA32B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 0, 0,2)
% Rojo verde azul para azul
DATA33B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 0, 0,3)
DATA34B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 0, 0,3)
DATA35B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 0, 0,3)
DATA36B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 0, 0,3)
% Rojo verde azul clear para rojo
DATA37B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 1, 0,1)
DATA38B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 1, 0,1)
DATA39B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 1, 0,1)
DATA40B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 1, 0,1)
% Rojo verde azul claro para verde
DATA41B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 1, 0,2)
DATA42B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 1, 0,2)
DATA43B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 1, 0,2)
DATA44B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 1, 0,2)
% Rojo verde azul clear para azul
DATA45B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 1, 1, 1, 0,3)
DATA46B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 1, 1, 1, 0,3)
DATA47B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 1, 1, 1, 0,3)
DATA48B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 1, 1, 1, 0,3)
% Rojo clear y cct para rojo
DATA49B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 1, 0, 0, 1, 1,1)
DATA50B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 1, 0, 0, 1, 1,1)
```

```
DATA51B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 1, 0, 0, 1, 1,1)
DATA52B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 1, 0, 0, 1, 1,1)
% Verde clear y cct para verde
DATA53B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 1, 0, 1, 1,2)
DATA54B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 1, 0, 1, 1,2)
DATA55B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 1, 0, 1, 1,2)
DATA56B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 1, 0, 1, 1,2)
% Azul clear y cct para azul
DATA57B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 0, 1, 1, 1,3)
DATA58B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 0, 1, 1, 1,3)
DATA59B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 0, 1, 1, 1,3)
DATA60B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 0, 1, 1, 1,3)
% CCT para rojo
DATA61B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 0, 0, 0, 1,1)
DATA62B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 0, 0, 0, 1,1)
DATA63B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 0, 0, 0, 1,1)
DATA64B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 0, 0, 0, 1,1)
% CCT para verde
DATA65B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 0, 0, 0, 1,2)
DATA66B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 0, 0, 0, 1,2)
DATA67B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 0, 0, 0, 1,2)
DATA68B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 0, 0, 0, 1,2)
% CCT para azul
DATA69B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen3,Datos_sen4, Color1,
Color2, Color3, Color4, 0, 0, 0, 0, 1,3)
DATA70B = analizaMARS(Datos_sen1,Datos_sen2,Datos_sen4,Datos_sen3, Color1,
Color2, Color4, Color3, 0, 0, 0, 0, 1,3)
DATA71B = analizaMARS(Datos_sen1,Datos_sen3,Datos_sen4,Datos_sen2, Color1,
Color3, Color4, Color2, 0, 0, 0, 0, 1,3)
DATA72B = analizaMARS(Datos_sen2,Datos_sen3,Datos_sen4,Datos_sen1, Color2,
Color3, Color4, Color1, 0, 0, 0, 0, 1,3)

% Se agrupan los datos
Datosregresion = [DATA1 DATA2 DATA3 DATA4 DATA5 DATA6 DATA7 DATA8 DATA9 DATA10
DATA11 DATA12 DATA13 DATA14 DATA15 ...
    DATA16 DATA17 DATA18 DATA19 DATA20 DATA21 DATA22 DATA23 DATA24 DATA25 DATA26
DATA27 DATA28 DATA29 DATA30 DATA31...
```

```
    DATA32 DATA33 DATA34 DATA35 DATA36 DATA37 DATA38 DATA39 DATA40 DATA41 DATA42
DATA43 DATA44 DATA45 DATA46 ...
    DATA47 DATA48 DATA49 DATA50 DATA51 DATA52 DATA53 DATA54 DATA55 DATA56 DATA57
DATA58 DATA59 DATA60 DATA61...
    DATA62 DATA63 DATA64 DATA65 DATA66 DATA67 DATA68 DATA69 DATA70 DATA71
DATA72]';
Datosspline = [DATA1B DATA2B DATA3B DATA4B DATA5B DATA6B DATA7B DATA8B DATA9B
DATA10B DATA11B DATA12B DATA13B DATA14B DATA15B ...
    DATA16B DATA17B DATA18B DATA19B DATA20B DATA21B DATA22B DATA23B DATA24B
DATA25B DATA26B DATA27B DATA28B DATA29B DATA30B DATA31B...
    DATA32B DATA33B DATA34B DATA35B DATA36B DATA37B DATA38B DATA39B DATA40B
DATA41B DATA42B DATA43B DATA44B DATA45B DATA46B ...
    DATA47B DATA48B DATA49B DATA50B DATA51B DATA52B DATA53B DATA54B DATA55B
DATA56B DATA57B DATA58B DATA59B DATA60B DATA61B...
    DATA62B DATA63B DATA64B DATA65B DATA66B DATA67B DATA68B DATA69B DATA70B
DATA71B DATA72B]';

% Se agrupan los datos recogidos para la representación de errores
total_elem = size(Datosregresion,1)/4;
Datosregresionb = zeros(total_elem,1);
Datossplineb = zeros(total_elem,1);

for n=1:total_elem
    for m=1:4
        Datosregresionb(n) = Datosregresionb(n)+Datosregresion((n-1)*4+m);
        Datossplineb(n) = Datossplineb(n)+Datosspline((n-1)*4+m);
    end
end

Datosregresionb = Datosregresionb/4;
Datossplineb = Datossplineb/4;

%% Spline + lineal multivariate
% Una vez que se puede discriminar cual es la mejor solución posible se
% hará la regresión completa por canal con los métodos evaluados. Con esto
% se consigue obtener las expresiones para hacer la regresión

% Canal rojo - spline todas las variables
nrow = size(Color,1);
alfa = ones(nrow,1);
Xmat_train = Datos_sens;
Xmat_train = [alfa,Xmat_train];
Ymat_train = Color(:,1);
params = aresparams(21, [], true, [], [], []);
model_red = aresbuild(Xmat_train, Ymat_train, params);
aresanova(model_red,Xmat_train,Ymat_train);

% Canal verde - spline todas las variables
alfa = ones(nrow,1);
Xmat_train = Datos_sens;
Xmat_train = [alfa,Xmat_train];
Ymat_train = Color(:,2);
params = aresparams(21, [], true, [], [], []);
model_green = aresbuild(Xmat_train, Ymat_train, params);
aresanova(model_green,Xmat_train,Ymat_train);
```

```
% Canal azul - lineal todas las variables
alfa = ones(nrow,1);
Xmat_train = Datos_sens;
Xmat_train = [alfa,Xmat_train];
Ymat_train = Color(:,3);
[b,bint,r,rint,stats] = regress(Ymat_train,Xmat_train);

%% Datos para comprobar
c1=[1 33101 18453   19452   34321   2346];
c2=[1 22303 19236   18356   25092   6386];
c3=[1 19337 16887   17800   20591   9749];
clc

rojo = arespredict(model_red,c3);
verde = arespredict(model_green,c3);
azul = b'*c3';
uint8(rojo)
uint8(verde)
uint8(azul)
%% Para el cálculo de la RED NEURONAL (con nftool)
% Esta parte del código permitirá hacer todas las iteraciones propuestas en
% el documento. Los parámetros que deberán ser modificados para que se
% pueda realizar de forma metódica serán hiddenLayerSize (para el número de
% neuronas en la capa oculta, net.trainFcn entre trainml y trainbr así como
% el nombre del archivo en el que guardar la red neuronal.

clc;
inputs = Datos_sens';
targets = Color';

% Se crea la red
hiddenLayerSize = 25;
net = fitnet(hiddenLayerSize);

% Se dividen los datos entre entrenamiento, validación y test
net.divideParam.trainRatio = 80/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 5/100;
net.trainFcn = 'trainlm';

% Se entrena la red
[net,tr] = train(net,inputs,targets);

% Se evalúa la red
outputs = net(inputs);
errors = gsubtract(outputs,targets);
performance = perform(net,targets,outputs)

% Se guarda la red
save('NN25_lm','net');
```

7.18. Anexo 18: Función para análisis de regresión lineal multivariable (script de MATLAB)

```
function DATA = analiza(train1,train2,train3,test, trainc1, trainc2, trainc3,
testc, var1, var2, var3, var4, var5, RGB)
% Matrices train para entrenamiento, test de test
% Los input var indicarán que variables habrá que tener en cuenta
% En A se devolverá el valor de R2 y el error cuadrático medio del grupo de
% test.
% RGB indicates 1 (red) 2 (green) 3 (blue)

% Variables a considerar de las 5 existentes
var = [];
if (var1==1)
    var = [var,1];
end
if (var2==1)
    var = [var,2];
end
if (var3==1)
    var = [var,3];
end
if (var4==1)
    var = [var,4];
end
if (var5==1)
    var = [var,5];
end


nrow = size(train1,1)+size(train2,1)+size(train3,1);
alfa = ones(nrow,1);

Xmat_train = [train1(:,var); train2(:,var); train3(:,var)];
Xmat_train = [alfa,Xmat_train];
Ymat_train = [trainc1(:,RGB); trainc2(:,RGB); trainc3(:,RGB)];

[b,bint,r,rint,stats] = regress(Ymat_train,Xmat_train);

% Se calcula el error cuadrático medio con los datos test
 alfa2 = ones(size(test,1),1);

 c = [alfa2, test(:,var)];
 datos = b'*c';

 error = datos'-testc(:,RGB);
%
 error2 = error.^2; % Error cuadrático
%
 error2medio = sum(error2)/size(testc,1);
 error2medio = sqrt(error2medio);

DATA = [error2medio];
```

7.19. Anexo 19: Función para análisis de regresión lineal multivariable (script de MATLAB)

```
function DATA = analizaMARS(train1,train2,train3,test, trainc1, trainc2,
trainc3, testc, var1, var2, var3, var4, var5, RGB)
% Matrices train para entrenamiento, test de test
% Los input var indicarán que variables habrá que tener en cuenta
% En A se devolverá el valor de R2 y el error cuadrático medio del grupo de
% test.
% RGB indicates 1 (red) 2 (green) 3 (blue)

% Variables a considerar de las 5 existentes
var = [];
if (var1==1)
    var = [var,1];
end
if (var2==1)
    var = [var,2];
end
if (var3==1)
    var = [var,3];
end
if (var4==1)
    var = [var,4];
end
if (var5==1)
    var = [var,5];
end


nrow = size(train1,1)+size(train2,1)+size(train3,1);
alfa = ones(nrow,1);

Xmat_train = [train1(:,var); train2(:,var); train3(:,var)];
Xmat_train = [alfa,Xmat_train];
Ymat_train = [trainc1(:,RGB); trainc2(:,RGB); trainc3(:,RGB)];

params = aresparams(21, [], true, [], [], []);
model = aresbuild(Xmat_train, Ymat_train, params);
aresanova(model,Xmat_train,Ymat_train);

% Se calcula el error cuadrático medio con los datos test

 error_sum = 0;
 for n=1:size(test,1)
 c = [1, test(n,var)];
 dato = arespredict(model,c);
 error = dato-testc(n,RGB);
 error2 = error.^2; % Error cuadrático
 error_sum = error_sum+error2;
 end
 error2medio = sum(error_sum)/size(testc,1);
 error2medio = sqrt(error2medio);
DATA = [error2medio];
```

7.20. Anexo 20: Extracción de colores de tarjeta (script de MATLAB)

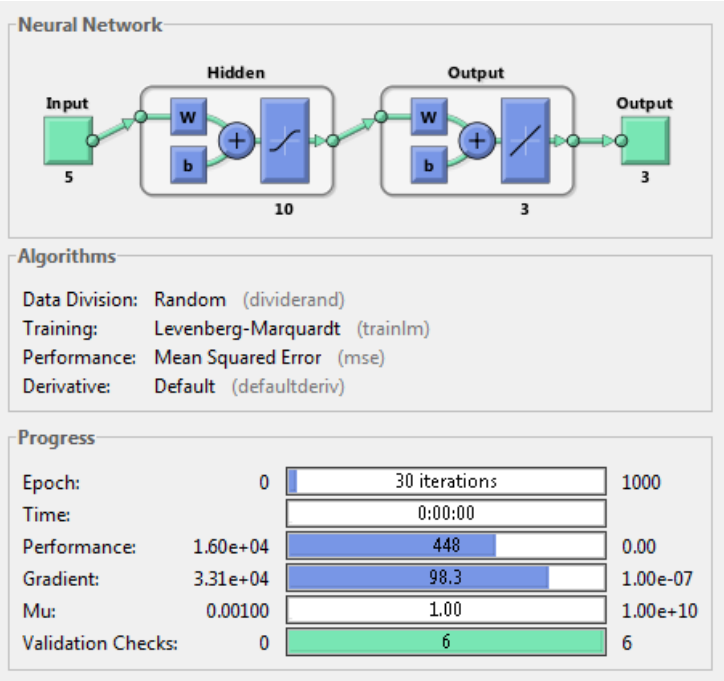
```
%% Programa de extracción de colores de tarjetas
% Permitirá la extracción del color medido sobre la tarjeta de calibración
% blanca que representará la fuente de luz. Con estos datos en sucesivos
% pasos se podrá realizar la calibración del sensor.
% El resultado será una matriz de 3 columnas que representará las
% componentes Red,Green,Blue; y tantas filas como número de imágenes de
% muestra existentes.

clear;
clc;
close all;
Color = [];

%% ========================================================================
% Final1
% ========================================================================
Im = imread('Final1.jpg'); % Se carga la imagen
% imagesc(Im); % Se muestra la imagen
% grid on; % Se muestra la rejilla
% Se busca el área de interés caracterizada por una columna de inicio y
% final y una fila de inicio y final que permitarán sacar un rectángulo de
% la imagen inicial para su análisis.
fil1 =1309;
fil2 =1542;
col1 =2539;
col2 =2861;
cur_row = fil2-fil1;
cur_col = col2-col1;
im_aux = Im(fil1:fil2,col1:col2,:);
% Se obtiene la media de la imagen capturada
RGBmean_aux=double(zeros(1,3));
for m=1:3
     for n=1:cur_row
       for o=1:cur_col
           RGBmean_aux(1,m)=RGBmean_aux(1,m)+double(im_aux(n,o,m));
       end
     end
   RGBmean_aux(1,m)=RGBmean_aux(1,m)/((fil2-fil1)*(col2-col1));
end
% La media que es una fila con los valores RGB como columnas se castea a
% formato entero de 8 bits sin signo
RGBmean_aux2=uint8(RGBmean_aux);
Color = [Color; RGBmean_aux2]

% Se representa el color obtenido
Tester = uint8(ones(100,100,3));
Tester(:,:,1) = Tester(:,:,1)*RGBmean_aux2(1);
Tester(:,:,2) = Tester(:,:,2)*RGBmean_aux2(2);
Tester(:,:,3) = Tester(:,:,3)*RGBmean_aux2(3);
figure;
imagesc(Tester);
```

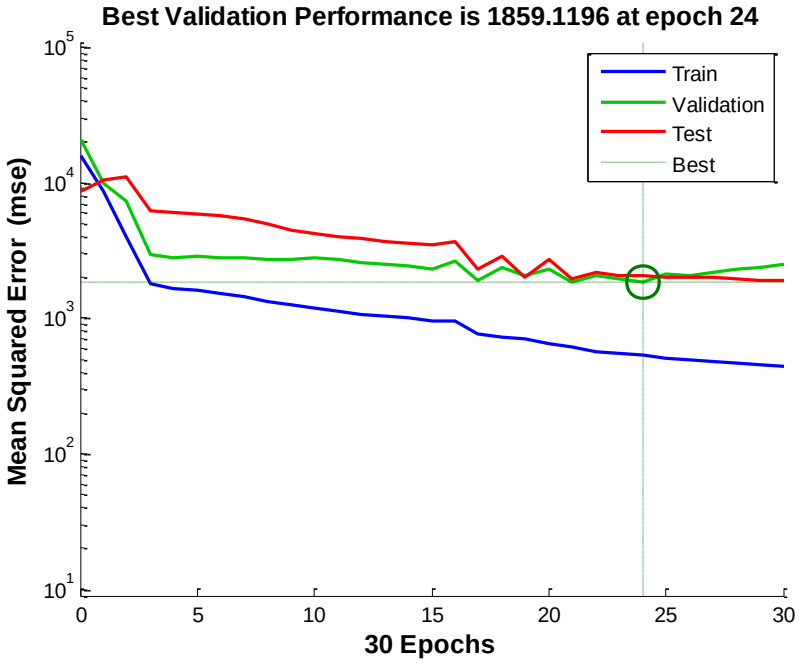
```
axis off;
title('Final1');
```

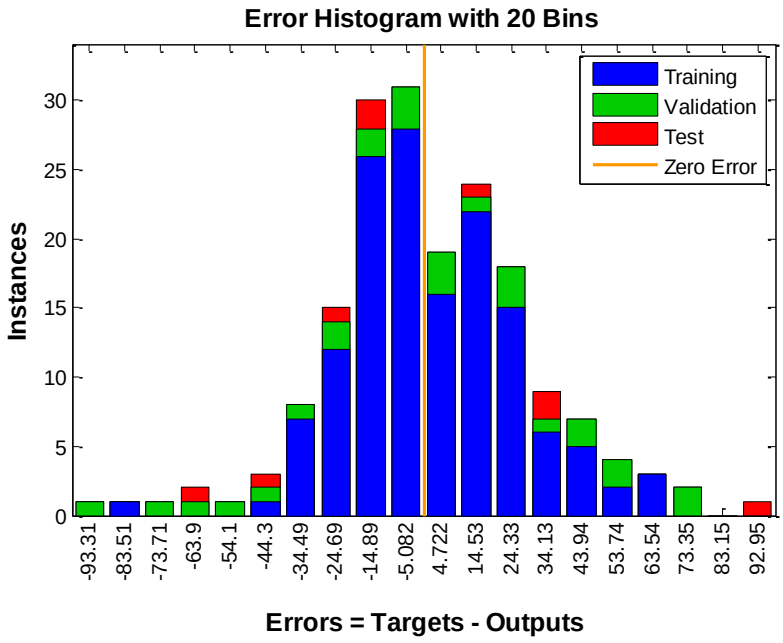
7.1. Anexo 21: Ensayos con Redes Neuronales

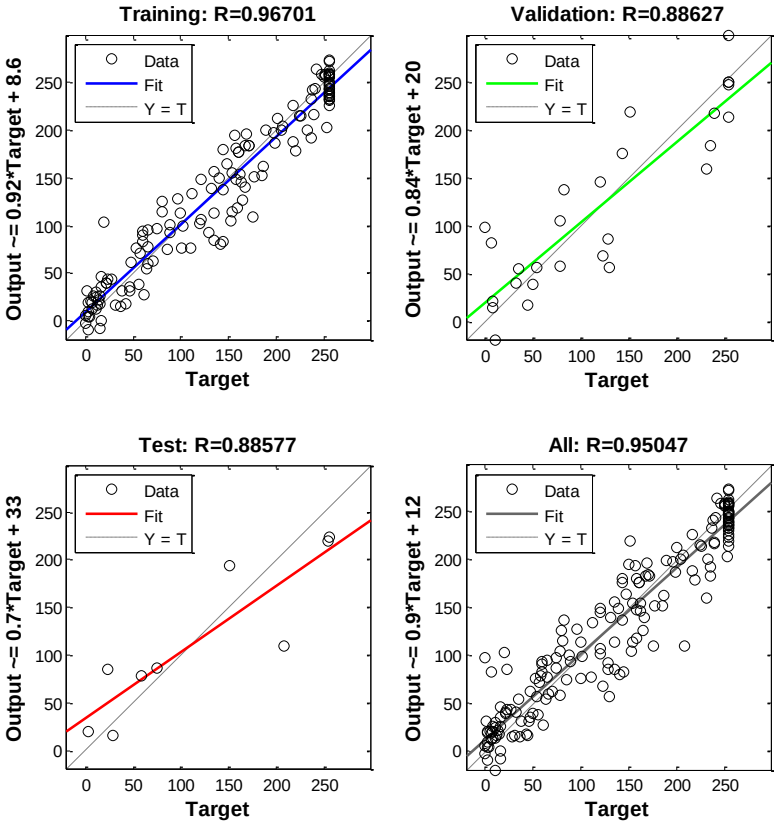
10 NEURONAS EN LA CAPA OCULTA

Levenberg-Marquardt

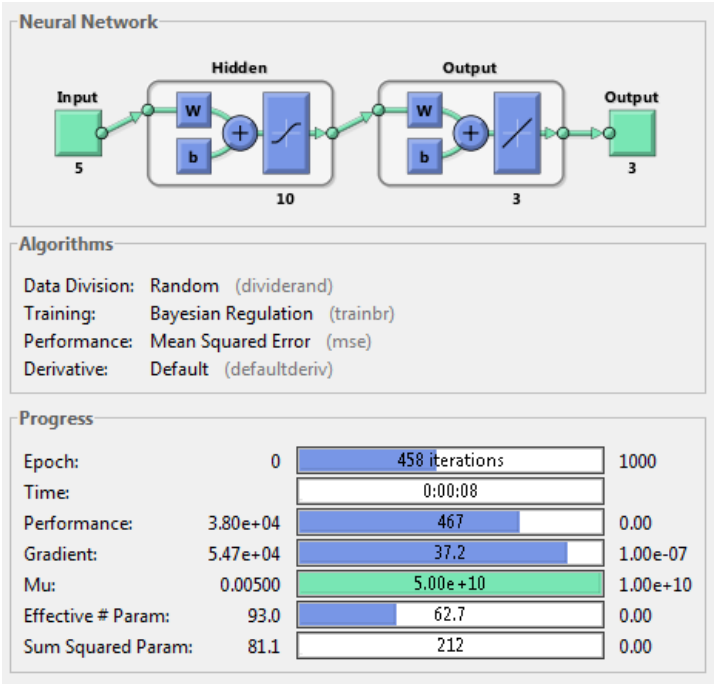


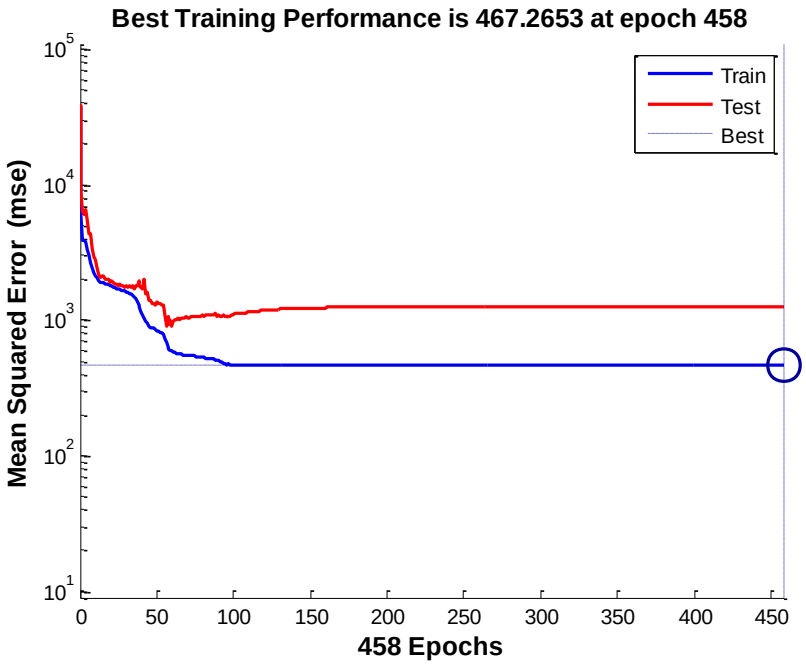

Best Validation Performance is 1859.1196 at epoch 24
Mean Squared Error (mse)
30 Epochs
Train
Validation
Test
Best

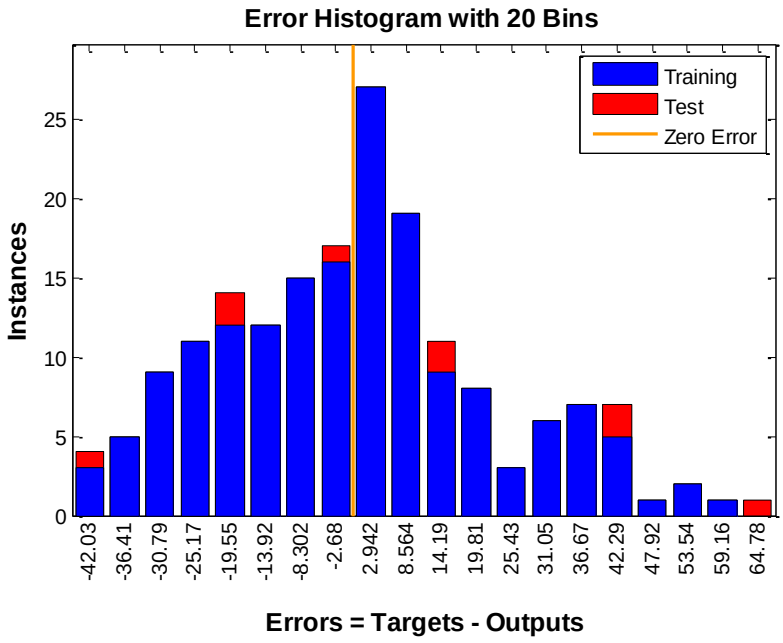

Error Histogram with 20 Bins
Instances
Errors = Targets - Outputs
Training
Validation
Test
Zero Error

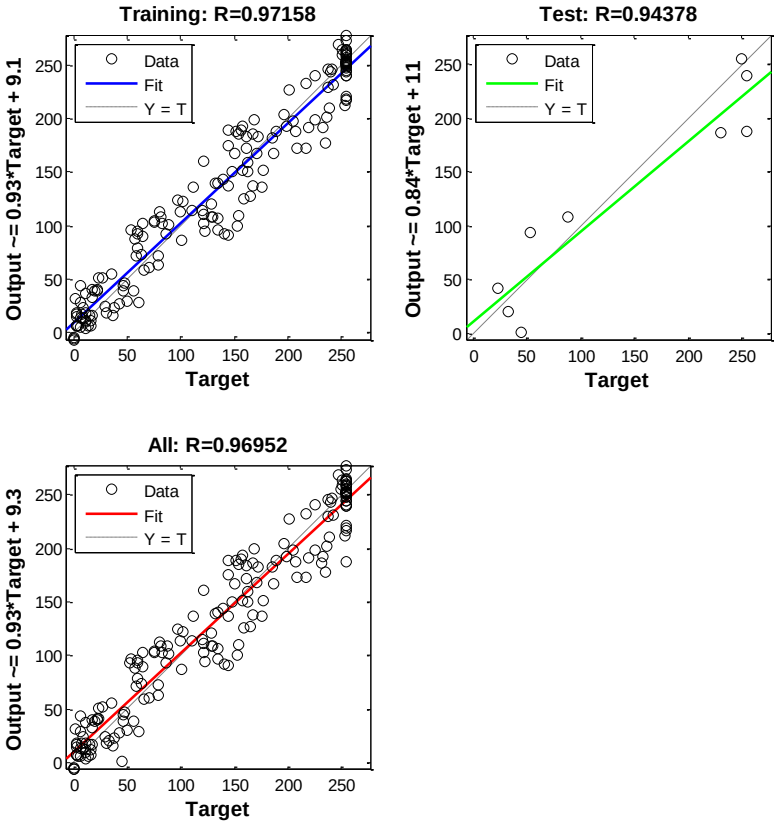


Levenberg-Marquardt con regularización Bayesiana



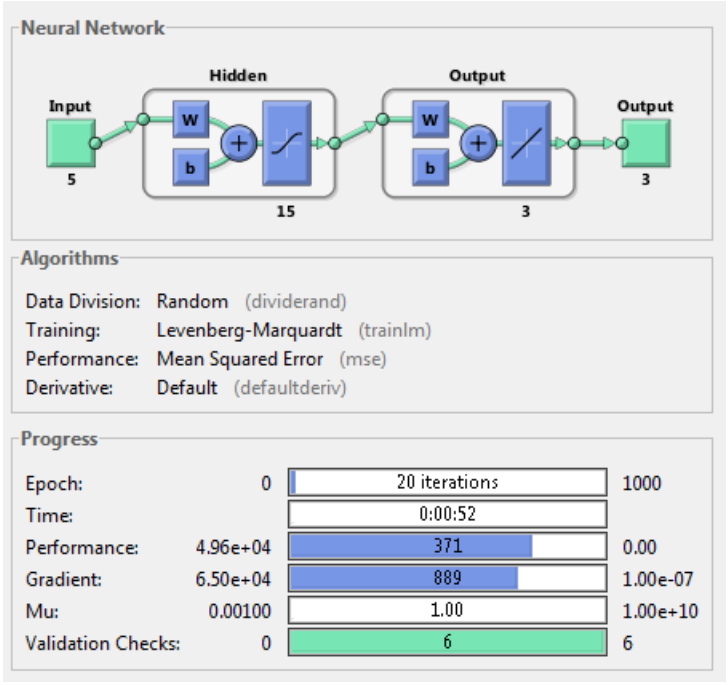

Best Training Performance is 467.2653 at epoch 458
Mean Squared Error (mse)
458 Epochs
Train
Test
Best

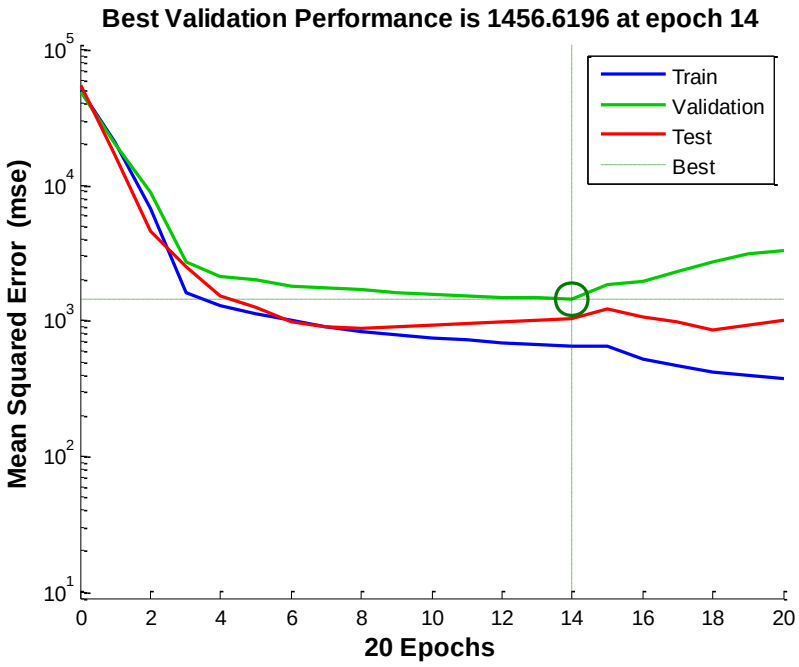

Error Histogram with 20 Bins
Instances
Errors = Targets - Outputs
Training
Test
Zero Error

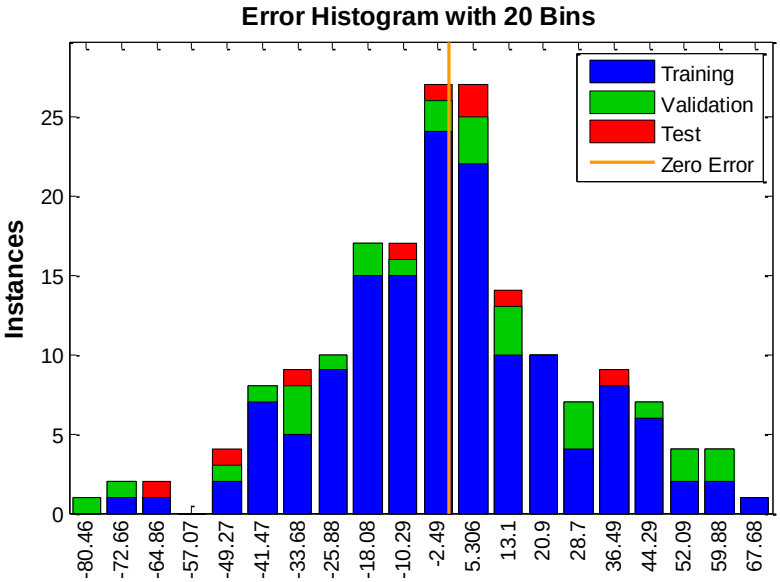


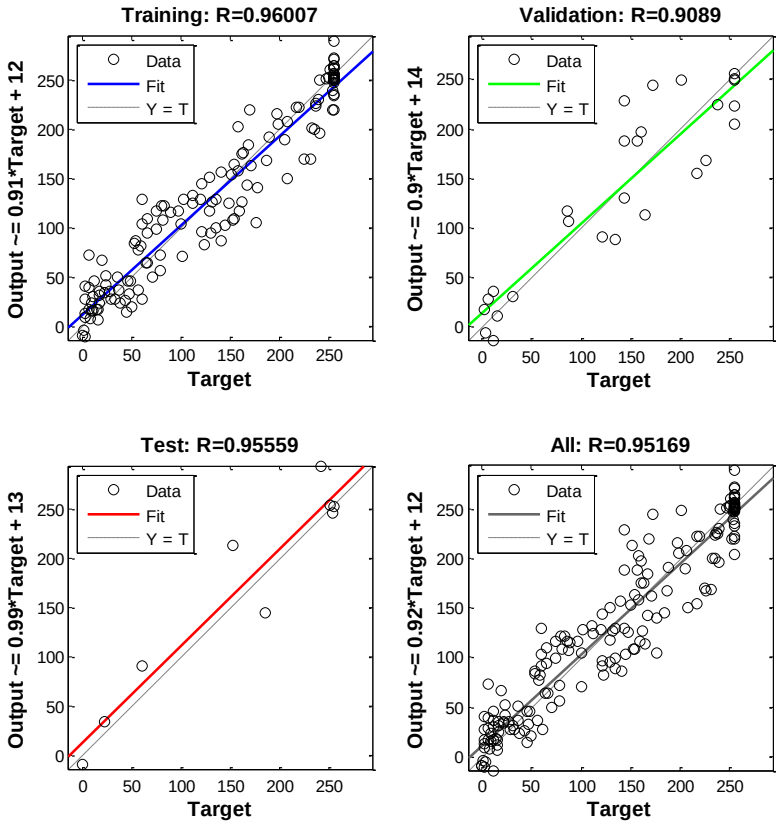
15 NEURONAS EN LA CAPA OCULTA

Levenberg-Marquardt

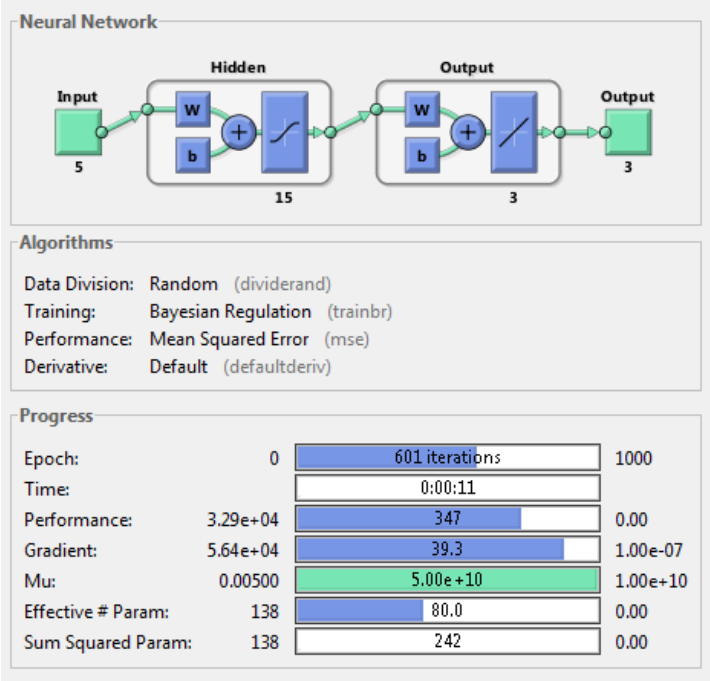

Neural Network
Input
5
Hidden
W
b
15
Output
W
b
3
Output
3
Algorithms
Data Division: Random (dividerand)
Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)
Progress
Epoch: 0 20 iterations 1000
Time: 0:00:52
Performance: 4.96e+04 371 0.00
Gradient: 6.50e+04 889 1.00e-07
Mu: 0.00100 1.00 1.00e+10
Validation Checks: 0 6 6

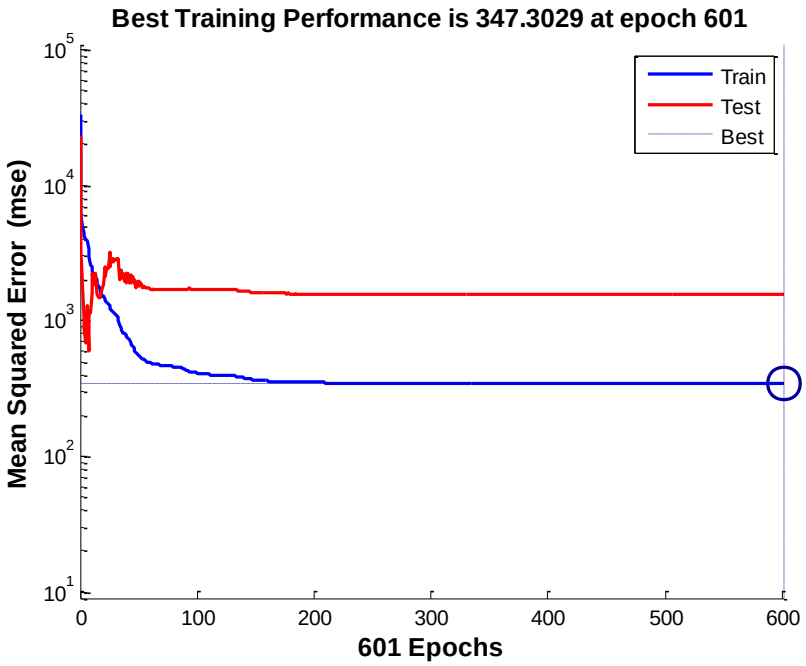

Best Validation Performance is 1456.6196 at epoch 14
Mean Squared Error (mse)
20 Epochs
Train
Validation
Test
Best

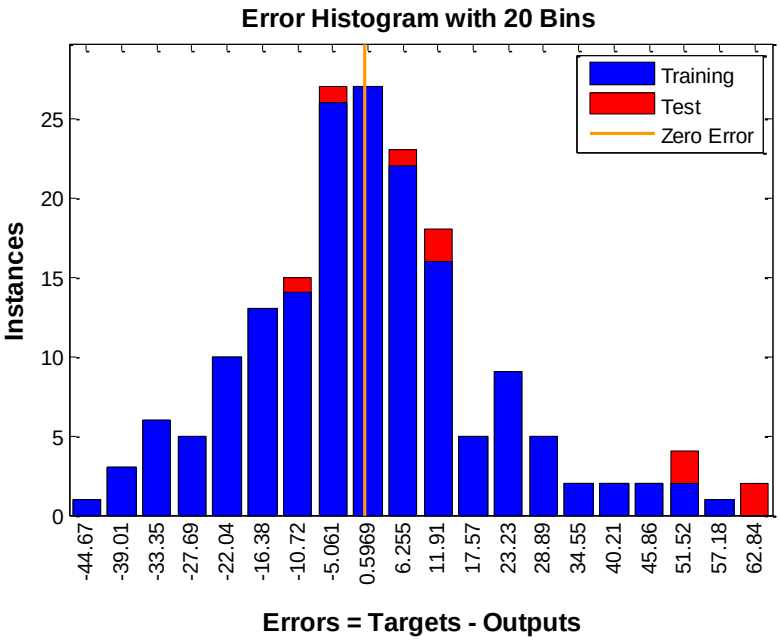


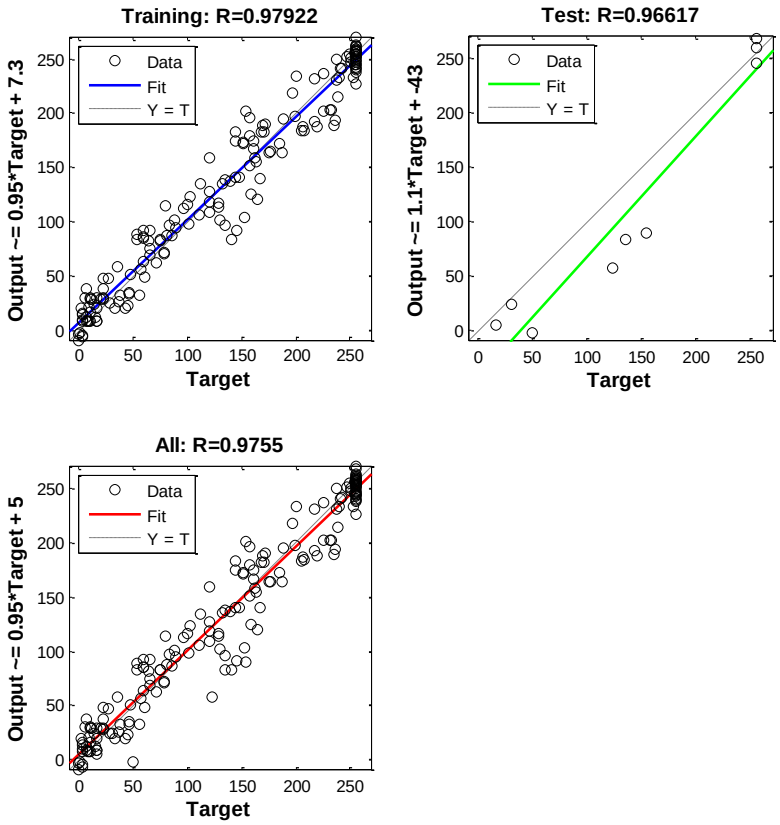


Levenberg-Marquardt con regularización Bayesiana


Neural Network
Input
5
Hidden
W
b
15
Output
W
b
3
Output
3
Algorithms
Data Division: Random (dividerand)
Training: Bayesian Regulation (trainbr)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)
Progress
Epoch: 0 601 iterations 1000
Time: 0:00:11
Performance: 3.29e+04 347 0.00
Gradient: 5.64e+04 39.3 1.00e-07
Mu: 0.00500 5.00e+10 1.00e+10
Effective # Param: 138 80.0 0.00
Sum Squared Param: 138 242 0.00

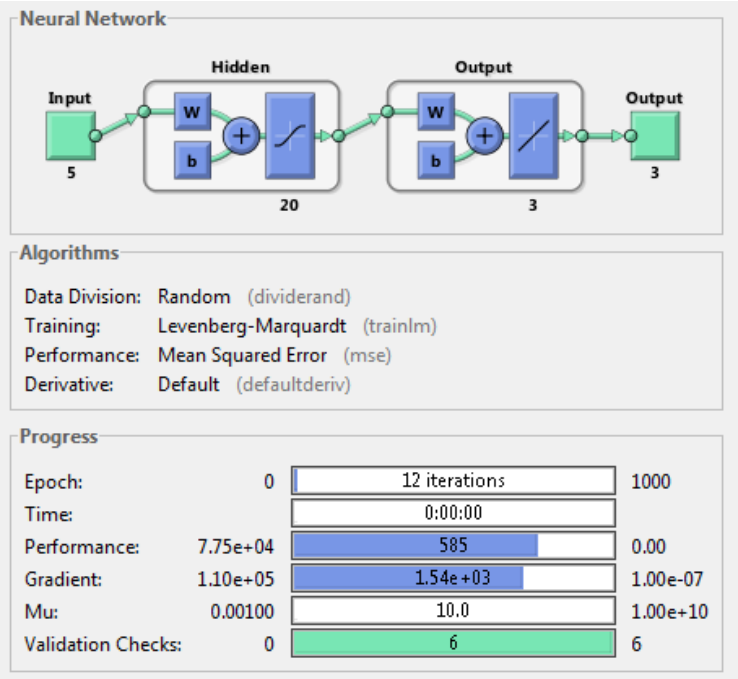

Best Training Performance is 347.3029 at epoch 601
Train
Test
Best
Mean Squared Error (mse)
601 Epochs
0
100
200
300
400
500
600

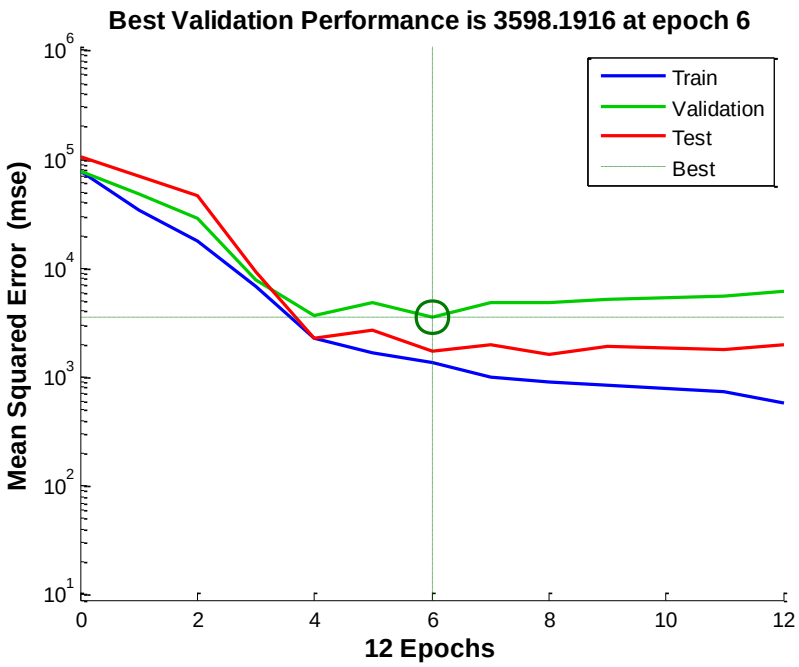

Error Histogram with 20 Bins
Training
Test
Zero Error
Instances
0
5
10
15
20
25
-44.67
-39.01
-33.35
-27.69
-22.04
-16.38
-10.72
-5.061
0.5969
6.255
11.91
17.57
23.23
28.89
34.55
40.21
45.86
51.52
57.18
62.84
Errors = Targets - Outputs

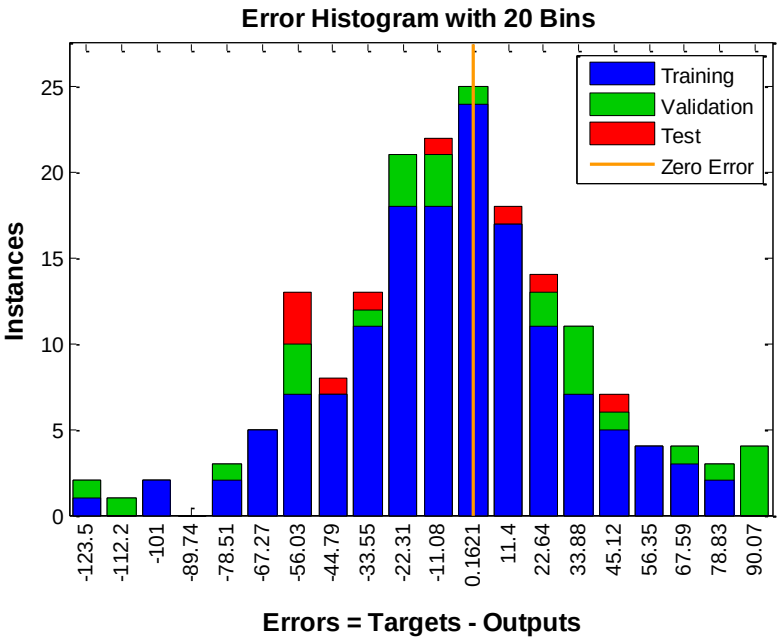

Training: R=0.97922
Data
Fit
Y = T
Output ~= 0.95*Target + 7.3
Target
Test: R=0.96617
Data
Fit
Y = T
Output ~= 1.1*Target + -43
Target
All: R=0.9755
Data
Fit
Y = T
Output ~= 0.95*Target + 5
Target

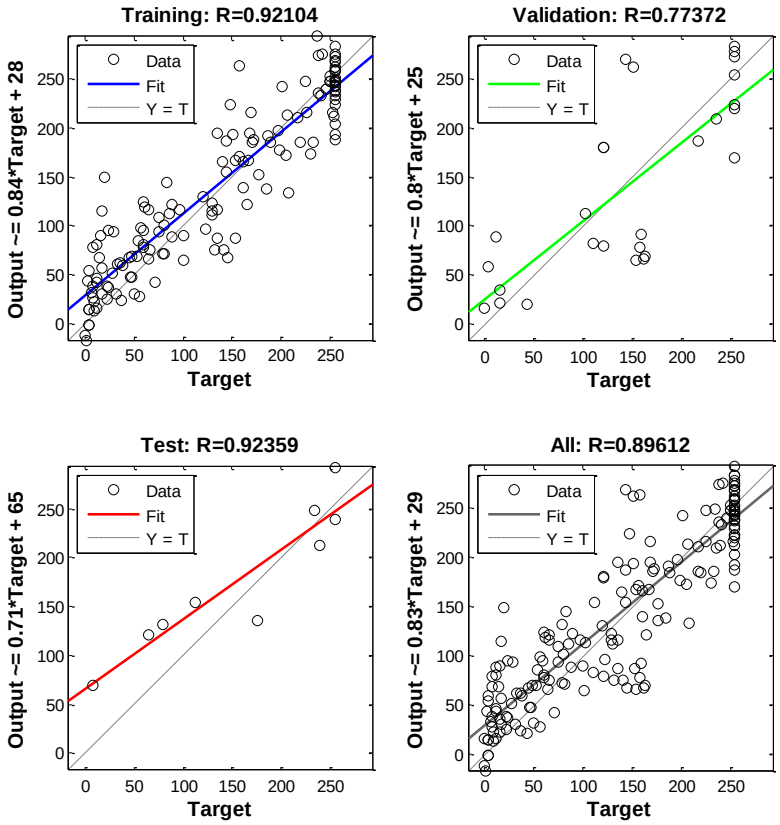
20 NEURONAS EN LA CAPA OCULTA

Levenberg-Marquardt

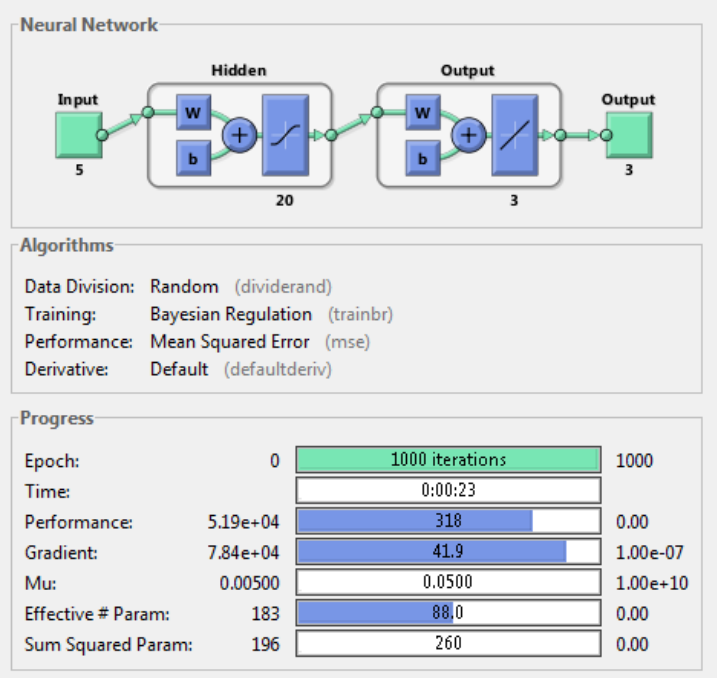


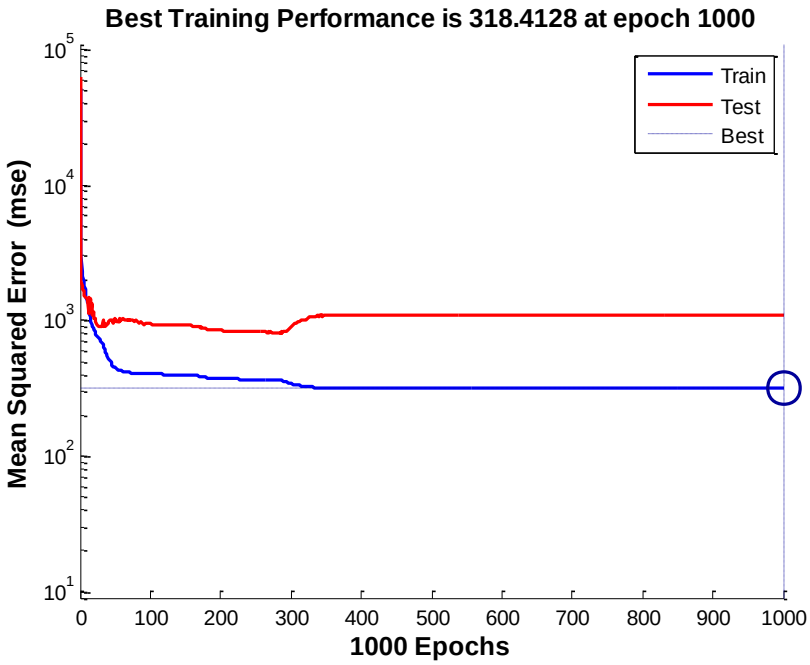


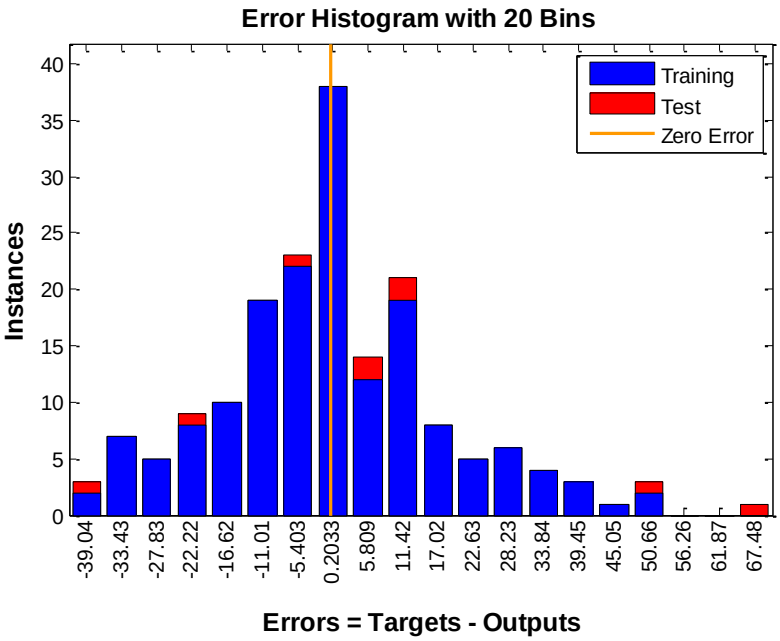

Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
25
20
15
10
5
0
-123.5
-112.2
-101
-89.74
-78.51
-67.27
-56.03
-44.79
-33.55
-22.31
-11.08
0.1621
11.4
22.64
33.88
45.12
56.35
67.59
78.83
90.07
Errors = Targets - Outputs

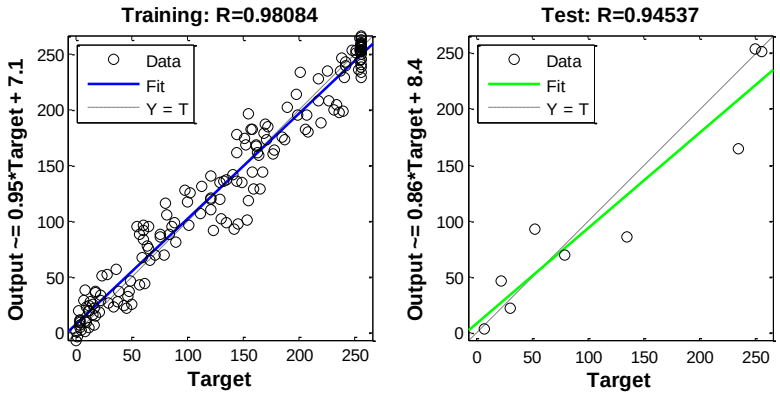

Training: R=0.92104
Data
Fit
Y = T
Output ~= 0.84*Target + 28
Target
Validation: R=0.77372
Output ~= 0.8*Target + 25
Test: R=0.92359
Output ~= 0.71*Target + 65
All: R=0.89612
Output ~= 0.83*Target + 29
0
50
100
150
200
250

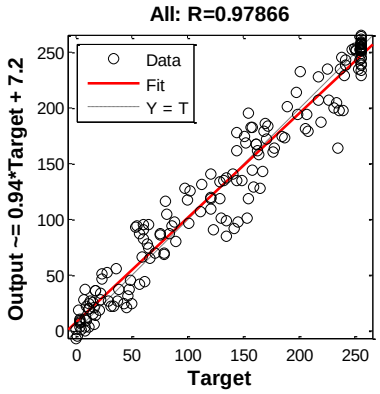
Levenberg-Marquardt con regularización Bayesiana





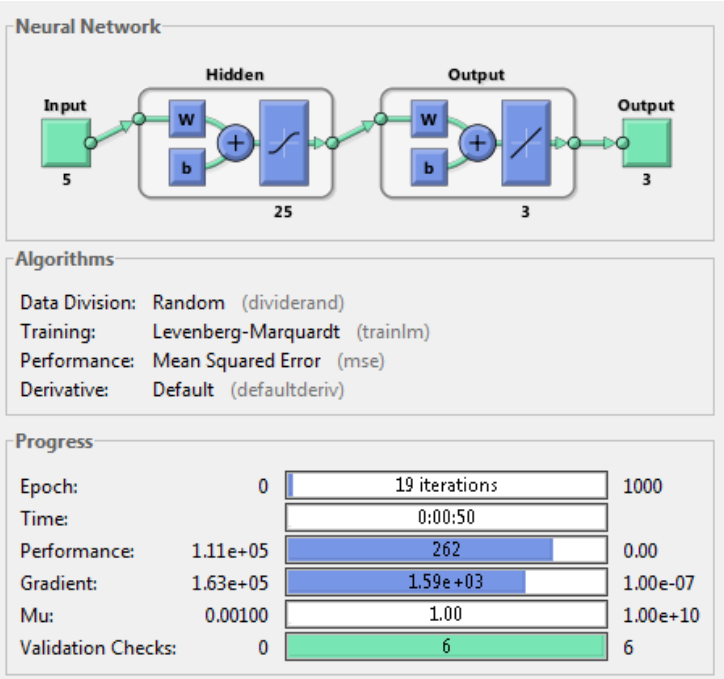

Error Histogram with 20 Bins
Training
Test
Zero Error
Instances
0
5
10
15
20
25
30
35
40
-39.04
-33.43
-27.83
-22.22
-16.62
-11.01
-5.403
0.2033
5.809
11.42
17.02
22.63
28.23
33.84
39.45
45.05
50.66
56.26
61.87
67.48
Errors = Targets - Outputs

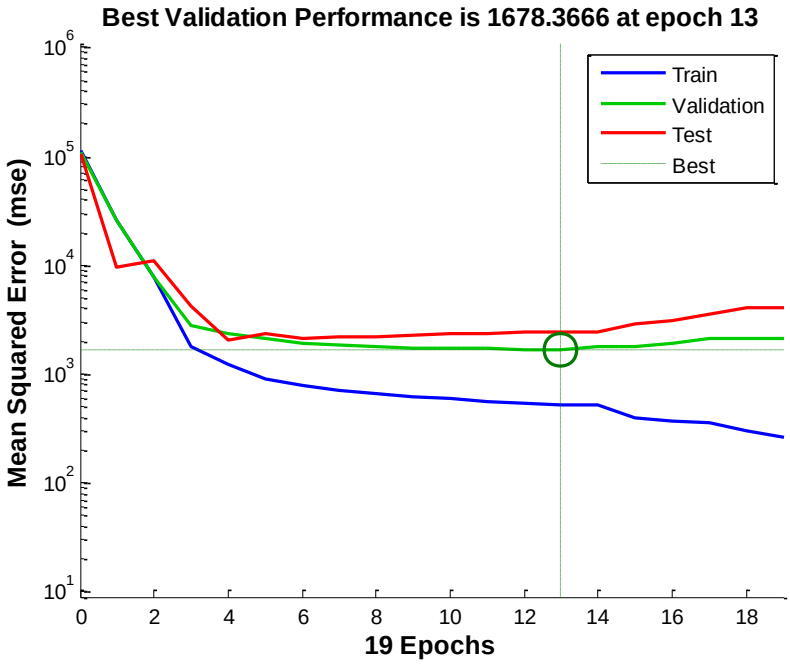

Training: R=0.98084
Data
Fit
Y = T
Output ~= 0.95*Target + 7.1
Target
Test: R=0.94537
Data
Fit
Y = T
Output ~= 0.86*Target + 8.4
Target

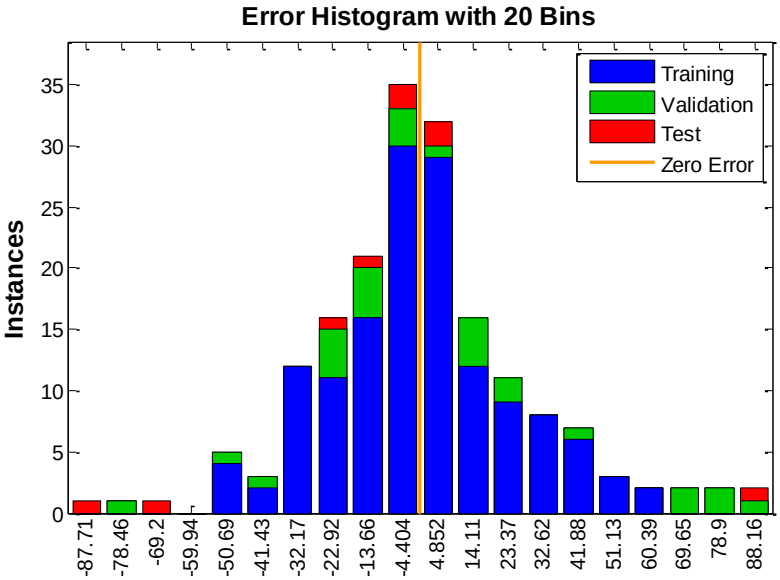

All: R=0.97866
Data
Fit
Y = T
Output ~= 0.94*Target + 7.2
Target

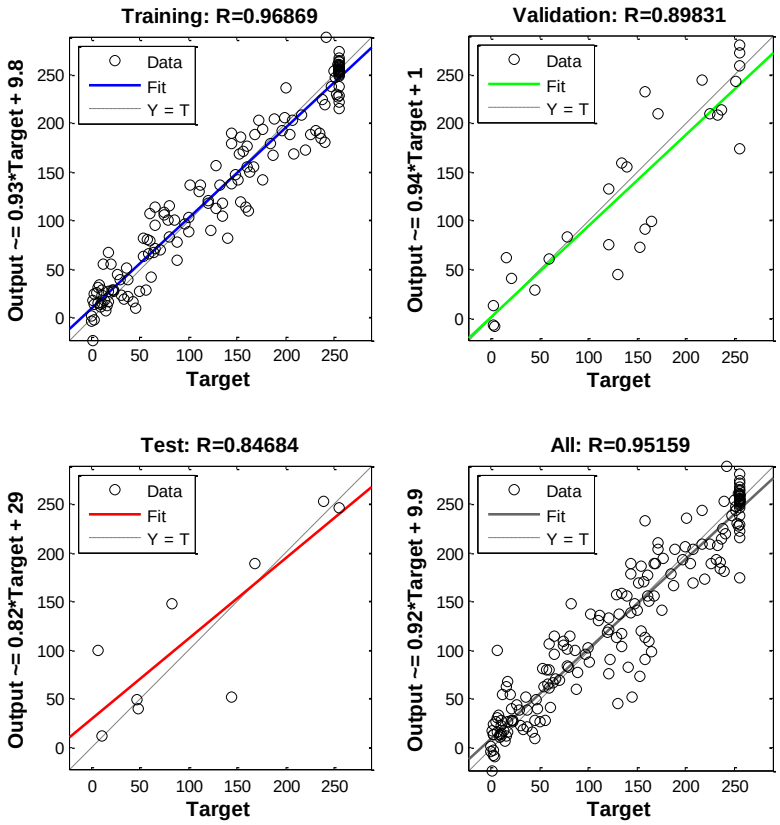
25 NEURONAS EN LA CAPA OCULTA

Levenberg-Marquardt

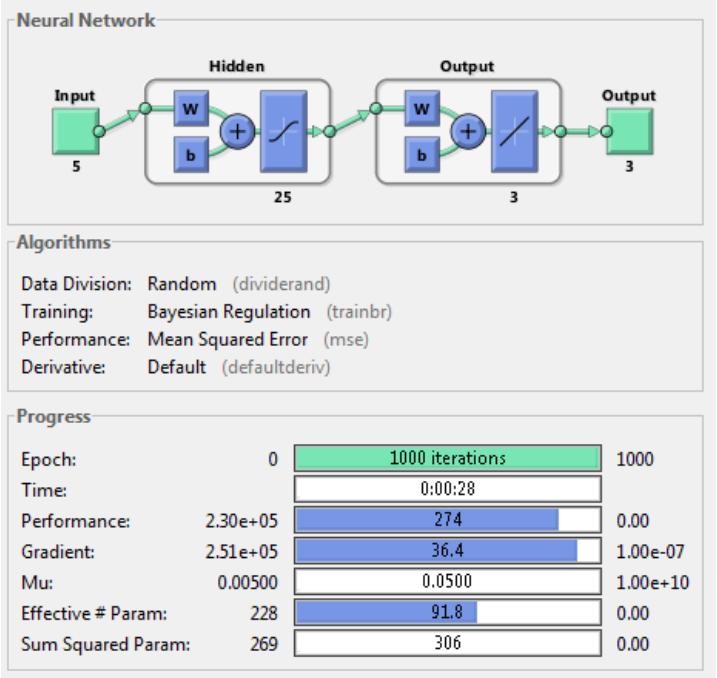


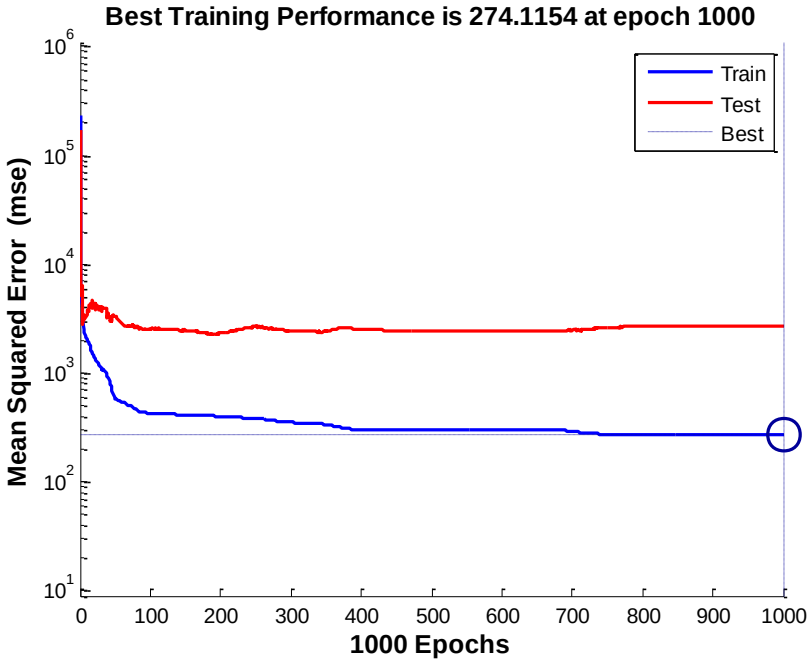


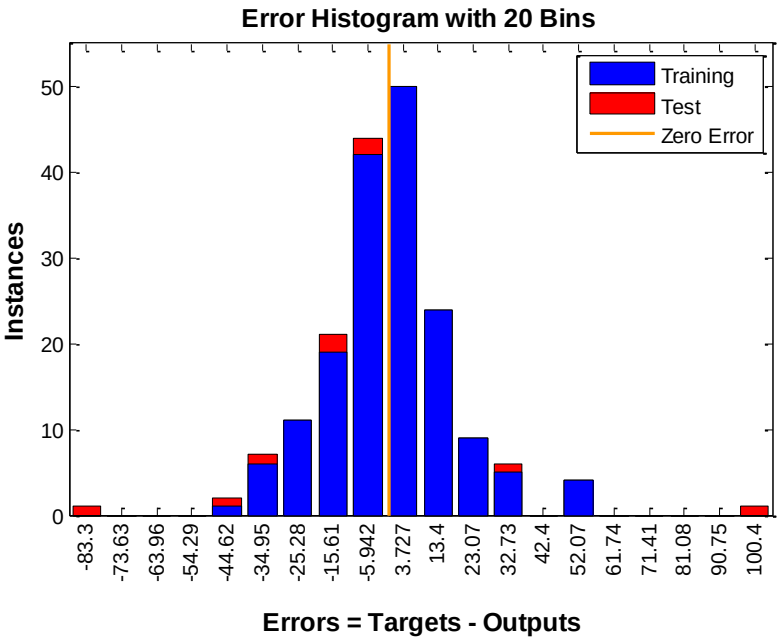

Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
-87.71
-78.46
-69.2
-59.94
-50.69
-41.43
-32.17
-22.92
-13.66
-4.404
4.852
14.11
23.37
32.62
41.88
51.13
60.39
69.65
78.9
88.16
Errors = Targets - Outputs

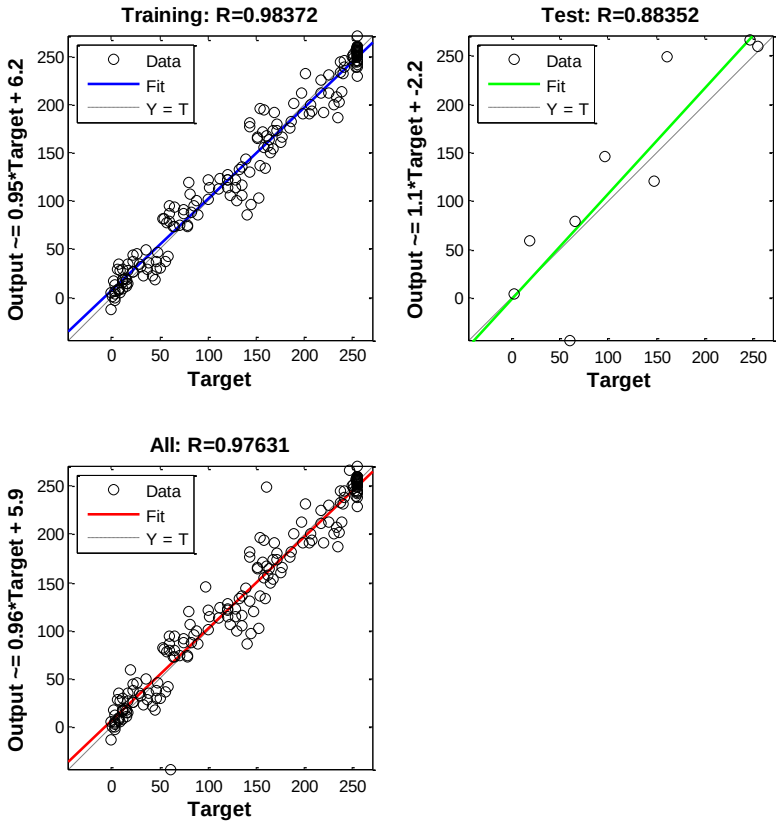

Training: R=0.96869
Data
Fit
Y = T
Output ~= 0.93*Target + 9.8
Target
Validation: R=0.89831
Output ~= 0.94*Target + 1
Test: R=0.84684
Output ~= 0.82*Target + 29
All: R=0.95159
Output ~= 0.92*Target + 9.9

Levenberg-Marquardt con regularización Bayesiana



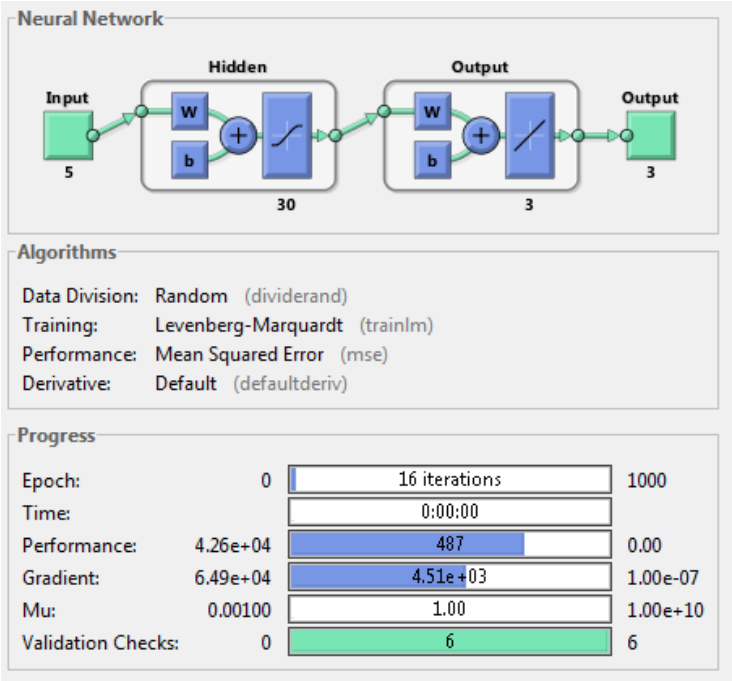


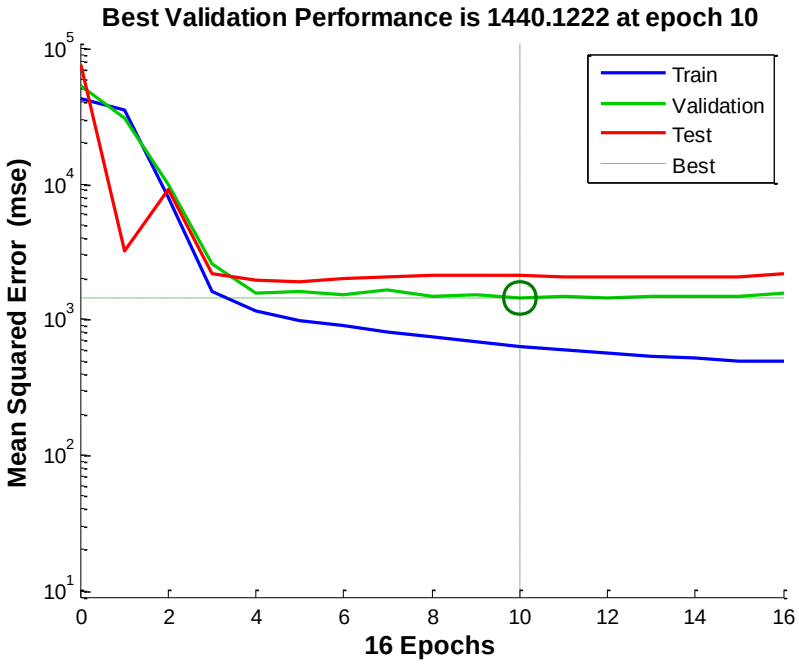

Error Histogram with 20 Bins
Training
Test
Zero Error
Instances
50
40
30
20
10
0
-83.3
-73.63
-63.96
-54.29
-44.62
-34.95
-25.28
-15.61
-5.942
3.727
13.4
23.07
32.73
42.4
52.07
61.74
71.41
81.08
90.75
100.4
Errors = Targets - Outputs

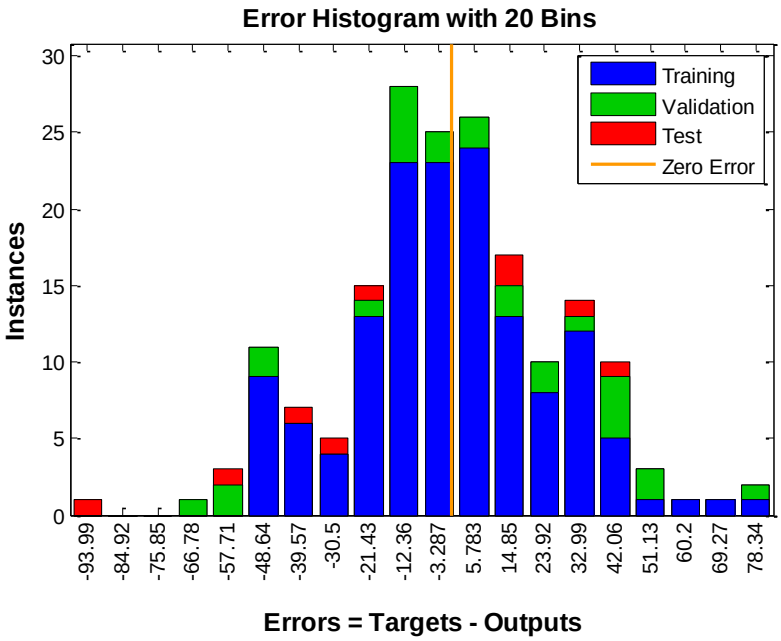

Training: R=0.98372
Data
Fit
Y = T
Output ~= 0.95*Target + 6.2
Target
Test: R=0.88352
Output ~= 1.1*Target + -2.2
All: R=0.97631
Output ~= 0.96*Target + 5.9
0
50
100
150
200
250

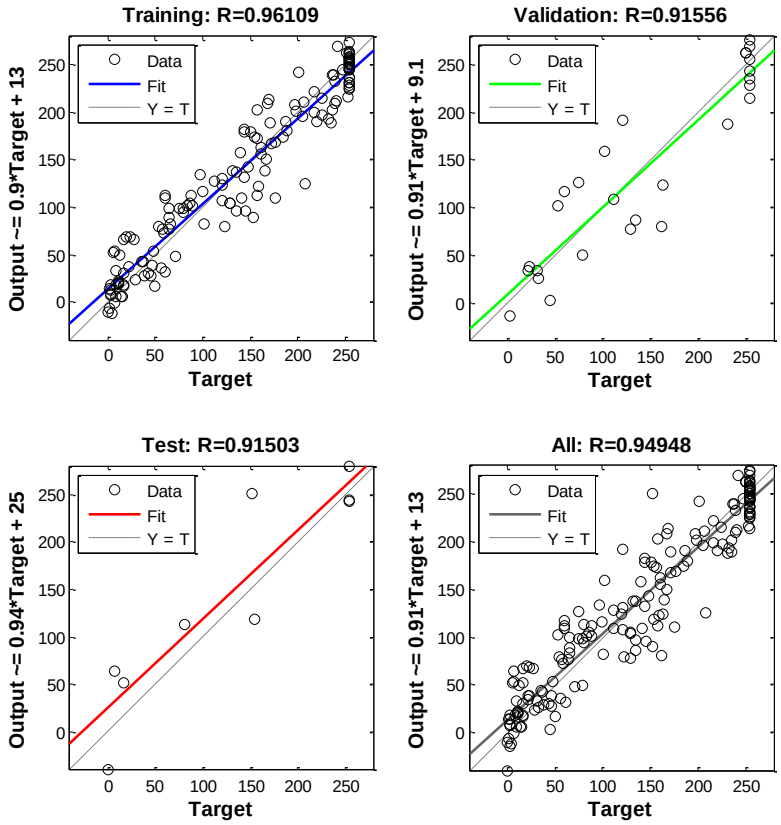
30 NEURONAS EN LA CAPA OCULTA

Levenberg-Marquardt

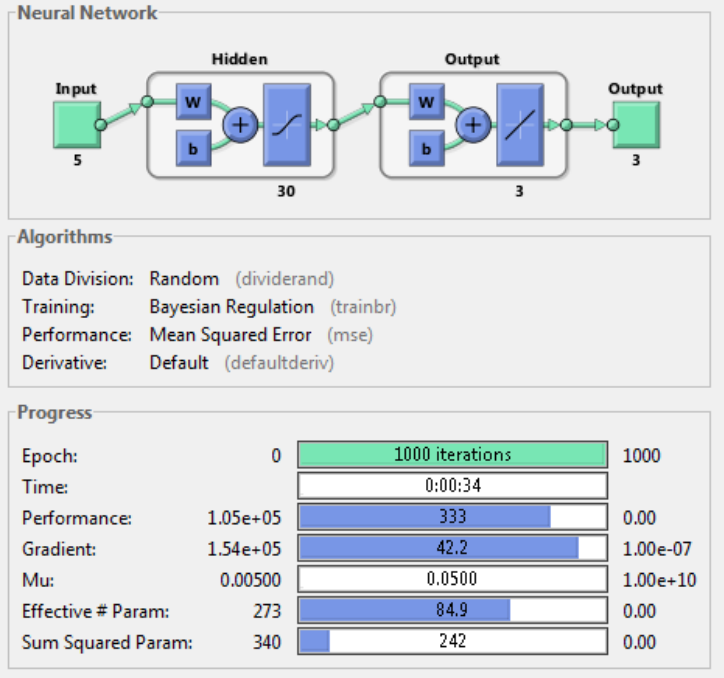


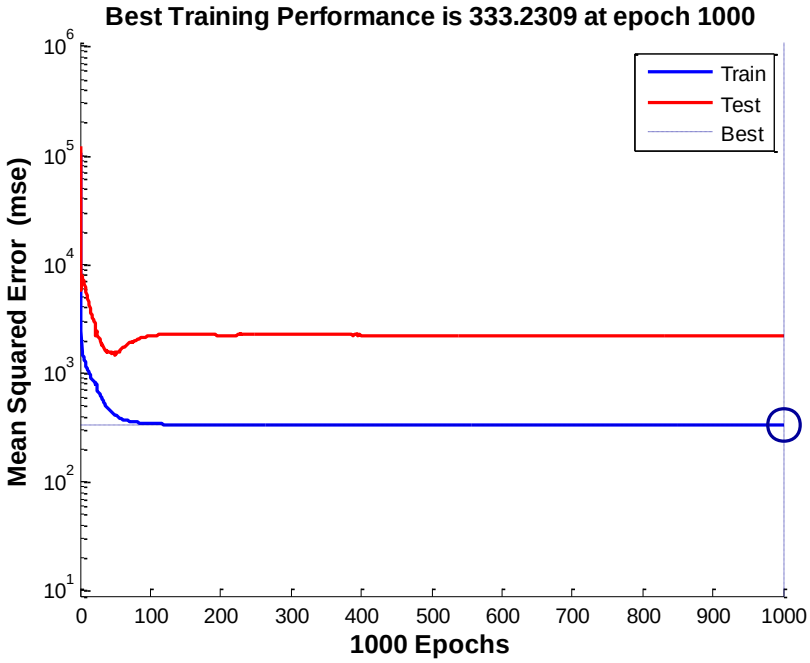


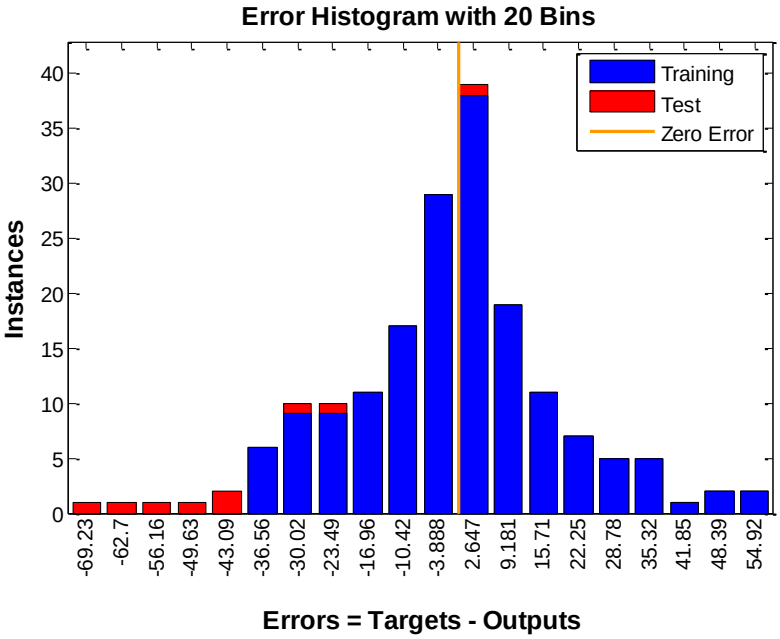

Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
30
25
20
15
10
5
0
-93.99
-84.92
-75.85
-66.78
-57.71
-48.64
-39.57
-30.5
-21.43
-12.36
-3.287
5.783
14.85
23.92
32.99
42.06
51.13
60.2
69.27
78.34
Errors = Targets - Outputs

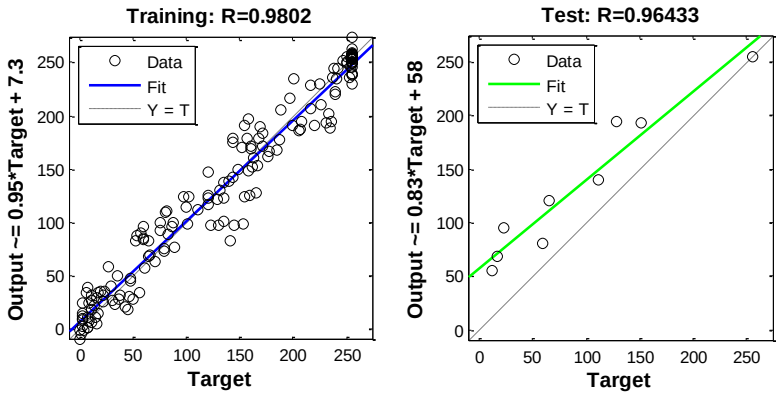

Training: R=0.96109
Data
Fit
Y = T
Output ~= 0.9*Target + 13
Target
Validation: R=0.91556
Output ~= 0.91*Target + 9.1
Test: R=0.91503
Output ~= 0.94*Target + 25
All: R=0.94948
Output ~= 0.91*Target + 13
0
50
100
150
200
250

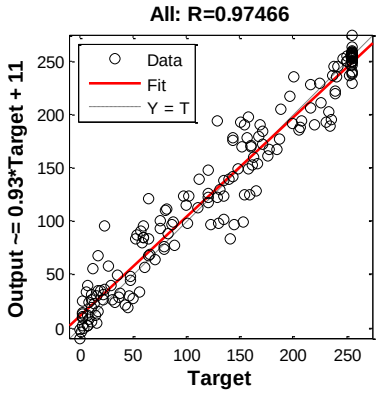
Levenberg-Marquardt con regularización Bayesiana






Error Histogram with 20 Bins
Training
Test
Zero Error
Instances
40
35
30
25
20
15
10
5
0
-69.23
-62.7
-56.16
-49.63
-43.09
-36.56
-30.02
-23.49
-16.96
-10.42
-3.888
2.647
9.181
15.71
22.25
28.78
35.32
41.85
48.39
54.92
Errors = Targets - Outputs


Training: R=0.9802
Data
Fit
Y = T
Output ~= 0.95*Target + 7.3
Target
Test: R=0.96433
Data
Fit
Y = T
Output ~= 0.83*Target + 58
Target


All: R=0.97466
Data
Fit
Y = T
Output ~= 0.93*Target + 11
Target

Las tablas siguientes muestran:

- El resultado de la aplicación de los anteriores modelos sobre el nuevo conjunto de imágenes.
- Los datos obtenidos en las tres primeras columnas los canales RGB obtenidos por la cámara y las tres siguientes los predichos por el algoritmo en base a los valores capturados por el sensor.
- Bajo cada tabla se mostrará un error que se corresponde con la suma de la desviación en valor absoluto de cada canal de color y cada test (3).

La nomenclatura de las tablas es la siguiente NNXXYY, donde:

- NN es Neural Network o Red Neuronal.
- XX indicará el número de neuronas en la capa oculta.
- YY puede ser LM o BR según sea respectivamente Levenberg-Marquardt o Levenberg-Marquardt con regularización Bayesiana

Tabla 13 – *NN10LM*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	255	45	79
Final3	172	246	49	132	202	31
Final4	7	139	255	26	68	224

Error = 351

Tabla 14 – *NN10BR*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	255	44	80
Final3	172	246	49	109	193	13
Final4	7	139	255	32	83	245

Error = 371

Tabla 15 – *NN15LM*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	255	37	90
Final3	172	246	49	112	230	33
Final4	7	139	255	26	114	255

Error = 261

Tabla 16 – *NN15BR*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	252	35	78
Final3	172	246	49	124	179	27
Final4	7	139	255	28	68	249

Error = 377

Tabla 17 – *NN20LM*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	255	27	76
Final3	172	246	49	100	226	15
Final4	7	139	255	27	110	255

Error = 324

Tabla 18 – *NN20BR*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	253	41	83
Final3	172	246	49	125	176	20
Final4	7	139	255	22	78	249

Error = 358

Tabla 19 – *NN25LM*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	251	46	62
Final3	172	246	49	105	226	0
Final4	7	139	255	33	80	255

Error = 369

Tabla 20 – *NN25BR*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	248	43	79
Final3	172	246	49	129	182	21
Final4	7	139	255	15	76	244

Error = 354

Tabla 21 – *NN30LM*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	255	43	101
Final3	172	246	49	117	239	10
Final4	7	139	255	24	100	255

Error = 265

Tabla 22 – *NN30BR*

Tipo	R	G	B	Rn	Gn	Bn
Final2	255	72	180	253	42	76
Final3	172	246	49	118	170	16
Final4	7	139	255	26	81	252

Error = 379