

UNIVERSIDAD POLITECNICA DE VALENCIA

ESCUELA POLITECNICA SUPERIOR DE GANDIA

Grado en Ing. Sist. de Telecom., Sonido e Imagen

---



UNIVERSIDAD  
POLITECNICA  
DE VALENCIA



ESCUELA POLITECNICA  
SUPERIOR DE GANDIA

**“Diseño e implementación de un  
algoritmo específico para la  
geolocalización de animales marinos:  
atunes”**

**TRABAJO FINAL DE GRADO**

Autor/a:

**Lluís Climent i Martí**

Tutor/a:

**José Pelegrí Sebastià**

**GANDIA, 2015**

# Abstract

El objetivo principal de este proyecto es implementar un algoritmo de geolocalización para el seguimiento de los patrones migratorios de animales marinos. El algoritmo puede resultar de especial interés en especies marinas que no salen prácticamente a la superficie, donde podrían utilizarse otros sistemas como el GPS, y en aquellas especies que realizan largas migraciones y requieren dispositivos con alta autonomía, pequeño tamaño, bajo peso y poco arrastre.

Así pues, se eligió la geolocalización por intensidad de luz ambiente, que requiere un sensor que capta la radiación de la luz solar y almacena los niveles de luz medidos con las respectivas fechas y horas. Estos datos se procesan posteriormente realizando una serie de cálculos astronómicos para obtener la latitud y la longitud del sensor en el momento en el que se tomó la medida. Dichos cálculos se realizan mediante el software GeoLight, basado en lenguaje R.

El sistema desarrollado es capaz de obtener una medida de coordenadas al día con un error de 53 km en longitud y de 44 km en latitud, lo cual para patrones migratorios que se pueden prolongar durante meses o años y a lo largo de miles de kilómetros resulta más que adecuado. También se planteó mejorar estos resultados añadiendo la geolocalización por campo magnético, aunque finalmente se desestimó por diversos factores.

This project's main objective was to implement an algorithm for migratory patterns of sea animals. This algorithm can be especially useful in marine species that rarely ascent to the sea surface where other systems such as GPS could not be used, and also species that perform long-run migrations and require long battery life, reduced size, low weight and little pull.

Therefore, light intensity geolocation was chosen, which requires a sensor that captures solar radiation and stores the measured light levels with their respective dates and hours. This data are processed afterwards performing some astronomical calculations to obtain the latitude and longitude where the sensor is located at the time it made the measure. These calculations were made over the software GeoLight, based in R language.

This system is able to obtain a coordinate measure per day, with an error of 53 km in longitude and 44 km in latitude, which with migratory patterns that may be extended over months or years, and through thousands of kilometers, is more than appropriate. Improving the results with geolocation by magnetic field was also an initial objective, but finally it was dismissed because of several factors.

**Palabras Clave:** Geolocalización, Animales marinos, Intensidad de luz solar, ecuaciones astronómicas, Geolight.

---

<b>1. INTRODUCCIÓN .....</b>	<b>4</b>
1.1 FUNDAMENTOS DEL GEOLIGHT .....	5
1.1.1 <i>Longitud</i> .....	6
1.1.2 <i>Latitud</i> .....	8
1.1.3 <i>Funcionamiento del algoritmo</i> .....	9
1.2 OBJETIVOS .....	11
1.3 METODOLOGÍA .....	12
1.3.1 <i>Intensidad de luz ambiente</i> .....	12
1.3.2 <i>Campo Magnético Terrestre</i> .....	14
1.3.3 <i>Temperatura de superficie marina</i> .....	16
1.4 SOLUCIÓN AL OBJETIVO.....	18
1.5 ESTRUCTURA DEL TRABAJO .....	18
<b>2. DESARROLLO .....</b>	<b>19</b>
2.1 PASOS PREVIOS .....	19
2.1.1 <i>Contextualización del proyecto</i> .....	19
2.1.2 <i>Planteamiento</i> .....	19
2.1.3 <i>Cambio de perspectiva: GeoLight</i> .....	21
2.2 SET-UP R Y GEOLIGHT .....	21
2.2.1 <i>Instrucciones de instalación de GeoLight</i> .....	22
2.2.2 <i>Utilización del GeoLight</i> .....	26
<b>3 SCRIPT GEOLLUIS.R .....</b>	<b>27</b>
3.1 CONSTRUCCIÓN DEL SCRIPT .....	27
3.1.1 <i>Código del Script y explicaciones pertinentes</i> .....	27
3.2 UTILIZACIÓN DEL SCRIPT.....	32
<b>4. RESULTADOS EXPERIMENTALES .....</b>	<b>36</b>
<b>5. CONCLUSIONES .....</b>	<b>41</b>
5.1 TRABAJO FUTURO .....	42
5.1.1 <i>Aplicaciones</i> .....	43
5.2 VALORACIÓN PERSONAL.....	44
5.3 AGRADECIMIENTOS.....	44
<b>6. REFERENCIAS .....</b>	<b>45</b>

# 1. Introducción

A lo largo de esta memoria se explicará porqué se ha optado por la utilización de la geolocalización por intensidad de luz ambiente para cumplir el objetivo del seguimiento de patrones migratorios de animales, así como el funcionamiento de dicha solución. También se explicará el enfoque que se le dio inicialmente al campo magnético y a la temperatura de la superficie del mar como mejoras de la precisión, y los motivos que llevaron a desestimar esta solución.

Se ha realizado este proyecto sobre un software llamado GeoLight [1], que es una librería del lenguaje de programación R dedicada a la geolocalización a partir de niveles de luz. Para explicar los fundamentos teóricos se tendrá como referencia la función “coord” [1] de la librería GeoLight, que es la encargada de obtener las coordenadas mediante complejos cálculos geométricos y astronómicos.

Para comprender cómo se realizan los cálculos necesarios para obtener la latitud y la longitud a partir de los niveles de luz, se deben dar unas nociones de coordenadas y astronomía. [2]

**Esfera celeste:** Plano esférico imaginario que engloba la tierra sobre el cual, aparentemente desde la tierra, se mueven los astros.

**Punto Aries:** Astro que sirve de punto de referencia en la esfera celeste para realizar cálculos de coordenadas.

**Ecuador celeste:** Plano paralelo al ecuador terrestre situado en la esfera celeste.

**Cénit:** Vertical de unión-intersección entre un punto en la tierra y un punto en la esfera celeste (p. ej. Un astro).

**Ascensión recta:** Ángulo medido sobre el ecuador celeste que forma el cénit del punto Aries con el de un astro.

**Tiempo Sidéreo:** Tiempo medido en ángulo de rotación de la tierra. Tiene como referencia el punto Aries y otro punto de la tierra, y 24 horas corresponden con 360 grados de rotación.

**GMST:** Greenwich Mean Sidereal Time, tiempo sidéreo medido entre el punto Aries y el meridiano de Greenwich.

**Ángulo Horario:** Ángulo medido sobre el ecuador que forma el cénit del sol con el meridiano de referencia del observador. En el caso del mediodía es de  $\pi$  radianes.

## 1.1 Fundamentos del GeoLight

A continuación se realizarán una serie de explicaciones de los fundamentos de la geolocalización por luz que realiza el GeoLight teniendo como referencia la función coord, tal y como se ha mencionado anteriormente.

El algoritmo del GeoLight utiliza una serie de aproximaciones probablemente basadas en libro "*Astronomy on the personal computer* [3]" de Oliver Montenbruck y Thomas Pfleger, investigadores del centro de operaciones espaciales de Alemania.

Este libro ofrece una serie de aproximaciones y herramientas para realizar cálculos sobre la mecánica celeste, como son determinar y predecir posiciones del sol, la luna, los planetas, cometas, eclipses, etc. mediante cálculos con software. En este caso se utilizan las aproximaciones de este libro (además de las ecuaciones de transformaciones entre sistemas de coordenadas [2]) para realizar los cálculos de la posición del sol respecto al horizonte, y obtener las coordenadas geográficas a partir de ésta.

A continuación se hace un resumen-relación del procedimiento que utiliza el algoritmo del GeoLight para el cálculo de las coordenadas según la fecha y la hora del amanecer y del anochecer, así como una breve explicación de los conceptos y parámetros utilizados para entender cómo se llega a calcular las coordenadas.

$Duracion\_dia = anochecer - amanecer;$

$Mediodía = amanecer + duracion\_dia/2$

Se establece la hora del mediodía tal y como marca la agencia "*National Oceanic and Atmospheric Administration*" (NOAA [4]), es decir, justo entre el amanecer y el anochecer. El sol realiza un recorrido a velocidad constante entre el horizonte al este y el horizonte al oeste. Dado que tomamos como umbral de un crepúsculo una cierta elevación del sol respecto al horizonte, y ésta es igual en el amanecer y el anochecer, es evidente que el sol alcanzará el punto más alto justo entre éstos dos fenómenos, independientemente del ángulo de elevación del sol que se tome como umbral. Éste hecho se puede observar claramente en la figura 1.

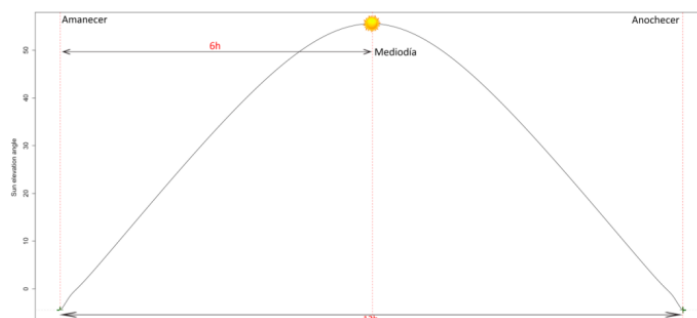


Figura 1 - Mediodía en una tabla de valores de luz

### 1.1.1 Longitud

El cálculo de la longitud se realiza a partir de la hora del mediodía.

*Mediodía-->Julian Date\*-->JC2000\**

Se cambia el formato del mediodía a fecha juliana y también a formato J2000. La fecha juliana es la cuenta de días desde la fecha de referencia del 1 de enero del 4713 AC a las 12 horas [5] hasta la fecha que se quiere obtener en este formato. Se eligió esta fecha como referencia para el formato de fecha juliana porque fue la última vez que coincidieron el ciclo solar, metónico e indicción.

El ciclo solar son 11 años, y es el período en el que el que cambia la actividad y apariencia solar [6].

El ciclo metónico son 19 años, y se utiliza porque es el mínimo común múltiplo entre años solares y meses lunares [7].

El ciclo indicción es de 15 años. Surgió en el antiguo Egipto, en el cual era solo de 5 años, periodo de las subidas del río Nilo. A lo largo de la historia se ha utilizado este período para distintos acontecimientos periódicos. Actualmente se utiliza datar documentos históricos [8].

A partir de la fecha juliana se obtiene la fecha en formato J2000, que tiene como referencia el 1 de enero del 2000 a las 12h.

Una vez cambiado el formato de fecha del mediodía, se realizan a partir de este, mediante las aproximaciones de la referencia [3], los siguientes cálculos:

*Oblicuidad de la elíptica*

*Anomalía media --> Longitud Eclíptica*

Y a partir de éstas:

*Oblicuidad* }  
                  } *Ascensión recta*  
*L\_Eclíptica* }

La oblicuidad de la elíptica [9] [2] es el ángulo que forma el eje de rotación de la tierra con la eclíptica, es decir, el eje de translación. Otra definición de la eclíptica sería el eje imaginario por el que el sol se desplaza de forma aparente alrededor de la tierra cuando el observador se encuentra en ésta. Esta definición de eclíptica se utiliza en las homónimas coordenadas para una mejor comprensión de éstas.

La anomalía media [10] [2] es un parámetro que da con la posición de un astro sobre la órbita a otro, en este caso la posición de la tierra sobre la elíptica. Se obtiene con la ley de Kepler.

La longitud eclíptica es la distancia, medida en ángulo a lo largo de la elíptica, con el sol. Este parámetro como su nombre indica se utiliza en las coordenadas elípticas [11] [2].

La ascensión recta, como se ha explicado anteriormente, es la longitud medida sobre el ecuador celeste (plano paralelo al ecuador de la tierra que corta con la esfera celeste) a partir del punto Aries o equinoccio vernal [12] [2]. En este caso se calcula la ascensión recta del sol.

Antes de calcular la longitud, se calcula el GMST [13] [2], que como ya se ha explicado es el tiempo sidéreo del sol respecto al meridiano de Greenwich.

Al tratarse de realizar el cálculo de la longitud en la que se encuentra el sensor utilizando como referencia la hora del mediodía, es evidente que el ángulo horario será de pi radianes. Finalmente se calcula la longitud con la siguiente ecuación astronómica:

$$\textit{Longitud} = \textit{Ascension Recta} - \textit{GMST} + \textit{Ángulo Horario}$$

La explicación a esta ecuación astronómica la encontramos en las figuras 2.a, 2.b y 2.c. [14]. En la figura 2.a vemos los 2 parámetros referenciados al punto Aries, el GMST y la Ascensión Recta.

Si a la ascensión recta le substraemos el ángulo del GMST obtenemos el ángulo que hay entre el meridiano de Greenwich y el cénit del sol, tal como observamos en la figura 2.b. Finalmente, al añadirle el Ángulo Horario del sol respecto al observador, como observamos en la figura 2.c obtenemos el ángulo entre el observador y el meridiano de Greenwich, más conocido como longitud.

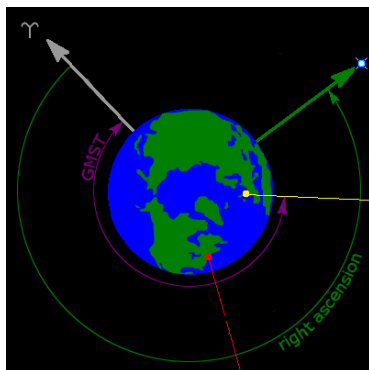


Figura 2.a

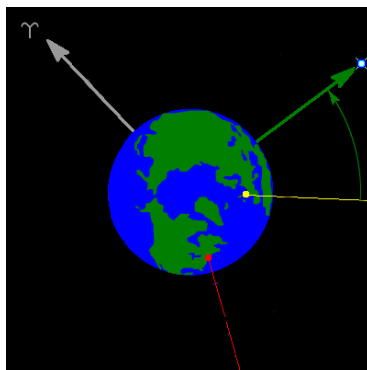


Figura 2.b

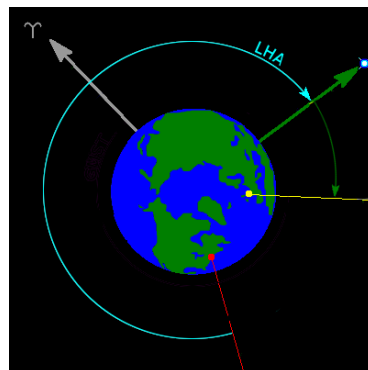


Figura 2.c

### 1.1.2 Latitud

Posteriormente se procede al cálculo de la latitud. Éste es el cálculo crítico debido a su mayor complejidad y a su mayor error, ya que depende de la duración del día, con lo cual una mala medida con el sensor o una mala calibración del umbral para detectar el amanecer y el anochecer, puede suponer cientos de kilómetros de error.

Se repite el proceso anterior, pero utilizando la hora del amanecer.

*Amanecer-->Julian Date\*-->JC2000\**

*Oblicuidad de la elíptica*

*Anomalía media --> Longitud Eclíptica*

Y a partir de éstas:

*Oblicuidad* }  
*L\_Eclíptica* } *Declinación (astronómica)*

La diferencia entre este cálculo y el anterior, es que se usan unas aproximaciones [3] distintas para calcular la Declinación (astronómica, no confundir con declinación magnética (apartado 1.3.2)) a partir del amanecer en lugar de la Ascensión Recta a partir del mediodía. Del mismo modo que la Ascensión Recta es el ángulo entre el punto Aries y el astro a lo largo del ecuador celeste, la Declinación es el ángulo entre el punto Aries y el astro a lo largo del círculo horario (plano paralelo al meridiano de Greenwich en la esfera celeste).



A continuación, a partir de una serie de ecuaciones astronómicas para la transformación entre sistemas de coordenadas, de las cuales se puede consultar el origen en el artículo científico "[Localization using Magnetometers and Light Sensors](#) [2]", se obtiene la latitud a partir de estos parámetros calculados.

Debido a la complejidad de los cálculos, las distintas operaciones se simplificarán y se agruparán en términos para su posterior tratado en el cálculo final.

$$term1 = \frac{\sin(elevación\_sol)}{\sqrt{\sin(declinación)^2 + (\cos(declinación) \cdot \cos(ángulo\_horario))^2}}$$

$$term2 = \frac{\cos(declinación) \cdot \cos(ángulo\_horario)}{\sin(declinación)}$$

Seguidamente se hace una aproximación condicional que depende de varios factores. Esta aproximación obtenida de la referencia [3], se puede observar también en el anexo, junto al resto del código de la función coord [1].

$$\left. \begin{array}{l} \text{Term1} \\ \\ \\ \text{Term2} \end{array} \right\} \left. \begin{array}{l} \sin^{-1}(term1) - \tan^{-1}(term2) \\ \sin^{-1}(term2) - \tan^{-1}(term2) \\ \cos^{-1}(term1) + \tan^{-1}\left(\frac{1}{term2}\right) \end{array} \right\} \text{Latitud}$$

Se trata de un cálculo condicional, es decir, que dependiendo de una serie de factores se realizará de una forma o de otra, por tanto es más complicado de lo que aparece en estos cálculos que se han simplificado para poder comprender a grandes rasgos cómo se calculan las coordenadas.

### 1.1.3 Funcionamiento del algoritmo

En la figura 3 se puede observar un diagrama de flujo que resume el cálculo de coordenadas, teniendo como referencia el software y hardware utilizado para realizar el proyecto.

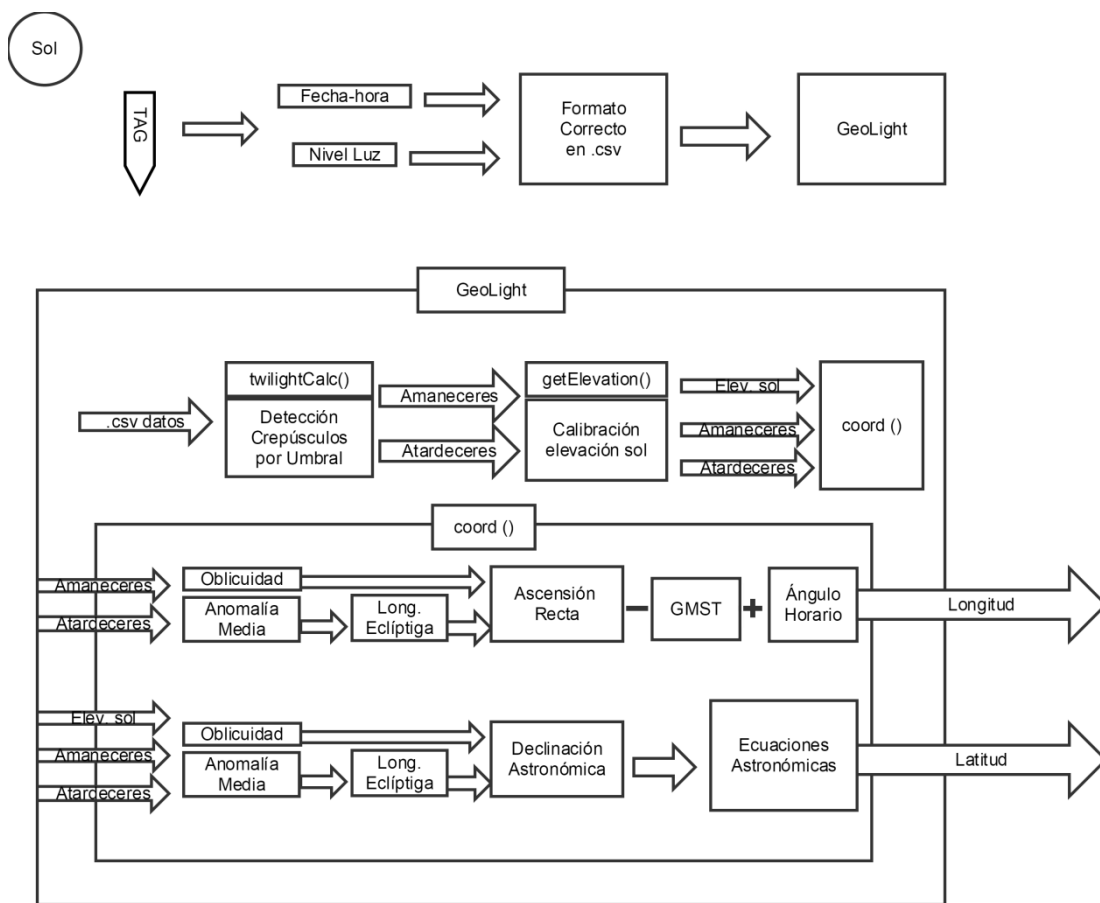


Figura 3 - Diagrama de Flujo Geolocalización con TAG y GeoLight

Como podemos observar, la luz del sol se capta mediante el sensor de luz del sensor o TAG (de éste se hablará en el apartado 2.1.2), el cual almacena los datos de luz asociados a su fecha y hora en una tabla.

Posteriormente se preparan los datos con el formato necesario en un archivo .csv, para introducirlo en el Script basado en la librería GeoLight desarrollado en este proyecto. Una vez introducidos los datos en el script, mediante las técnicas que se acaban de describir se obtienen las coordenadas.

## 1.2 Objetivos

Los objetivos iniciales de este proyecto son los siguientes:

- Diseñar un algoritmo que sea capaz de geolocalizar a un atún.
- Implementar el algoritmo en una plataforma software para poder ser utilizado en casos reales.
- Testear el algoritmo con datos registrados de la luz solar y la temperatura del mar.

Como se ve puede observar, el objetivo principal de este trabajo de fin de grado es realizar un sistema de geolocalización que sea capaz de obtener de la forma más eficiente posible las coordenadas en las que se encuentra el animal marino, concretamente un atún rojo a lo largo de su migración.

El atún rojo realiza su migración por el mediterráneo, saliendo a mar abierto por el estrecho de Gibraltar, y recorriendo miles de kilómetros a lo largo del atlántico (figura 4).



Figura 4 - Patrón migratorio del atún rojo [15]

## 1.3 Metodología

Para obtener y optimizar los resultados requeridos se han contemplado diversas soluciones y mejoras para el algoritmo, entre las cuales destacan la geolocalización por intensidad de luz ambiente, por campo magnético terrestre, y por temperatura de superficie marina.

### 1.3.1 Intensidad de luz ambiente

Se propone utilizar la geolocalización por intensidad de luz ambiente. Este método de geolocalización utiliza un sensor de luz de muy reducidas dimensiones y peso (similares a una tarjeta de memoria SD) y que requiere de muy poca energía para su funcionamiento.

Así pues, se debe implementar un algoritmo que sea capaz de localizar a partir de datos de intensidad de luz ambiente en una plataforma software para poder ser utilizado en casos reales.

Este sistema se basa en medidas de radiación solar tomadas a lo largo del día, cada una de las cuales lleva asociada en un formato de tabla la fecha y la hora en la que fue tomada. La localización de dichas medidas se realiza a posteriori mediante un procesamiento de los datos tomados por el sensor, que mediante una serie de cálculos, aproximaciones y ecuaciones astronómicas es capaz de obtener la latitud y la longitud en la que se encontraba el sensor en el día que se realizó la medida [2] [3].

Según los datos obtenidos en investigaciones previas, este sistema es capaz de dar las coordenadas con un error medio de entre 50 km y 100 km en longitud, y de entre 150 km y 250 km en latitud en la mayoría de sistemas existentes en la actualidad [16] [17].

Una vez se haya realizado la implementación de este algoritmo se testeará con datos reales de luz solar y temperatura del mar registrados mediante un dispositivo multi-sensor o TAG desarrollado por el Grupo de Sensores y Magnetismo del Instituto de Investigación para la Gestión Integrada de Zonas Costeras (GSyM del IGIC), del cual se hablará en el apartado 2.1.2.

Respecto a la frecuencia de medidas, se debe mencionar que se obtiene una sola medida de localización al día, dado que para obtener las coordenadas a partir de las medidas de luz se utilizan las horas de amanecer, anochecer y mediodía que obviamente sólo se dan una vez cada 24 horas.

La latitud tiene un mayor error ya que, como ya se ha mencionado en el apartado 1.1.2, depende de las horas de luz solar que se han medido en el día calculado, lo cual depende de las horas a las que se han captado el amanecer y el anochecer que pueden tener diversas fuentes de error.

Estas fuentes de error pueden ser debido al hardware, por una mala medida en el animal debido a una mala orientación con respecto al sol en esos momentos, o un movimiento inusual del aparato como vibraciones continuas.

También pueden darse errores por el procesado de datos, ya que para obtener las horas del amanecer y del anochecer se utilizan ciertos umbrales de luz los cuales si no se calibran bien pueden llevar a “falsos positivos” que conllevan excesivas o escasas horas de luz solar que en casos extremos pueden llegar a ser claramente inverosímiles (por ejemplo 3 horas de luz solar en el mediterráneo).

En cambio, la longitud es en cierto modo invulnerable a estos errores dado que depende de la hora del mediodía la cual se mantiene inalterada independientemente de los errores de umbral en el amanecer y el anochecer. Esto se ve de forma más clara y evidente en la figura 1.

Por mucho que se suba o se baje el umbral debido a una mala calibración el mediodía sigue situado en el mismo punto, justo el punto intermedio entre el amanecer y el anochecer (cálculos según el NOAA [4]).

En cambio, si el error se debiera a una mala medida por parte del sensor a causa de los factores anteriormente mencionados, la longitud sí que se vería afectada, ya que los “falsos positivos” conllevarían a captar amaneceres o anohecidos falsos y desplazados irregularmente (por ejemplo, amanecer a las 6:00 AM y anochecer a las 7:30 AM), lo cual derivaría en horas del mediodía inverosímiles y por tanto en un error en longitud.

Así pues, identificar si el error proviene de una mala calibración de los umbrales de detección del amanecer y el anochecer (entre otros que se mencionarán más adelante) o de una mala medida por el hardware se podría identificar viendo si se ha producido un error en la longitud, o si el error sólo viene de la latitud.

En medidas de calibración donde se conoce la localización del sensor se identificarían fácilmente los errores, pero en una medida en la que se desconoce la localización real, como cuando el sensor está montado en el animal, esto se podría ver según las medidas contextuales, es decir, si la medida que se está observando para identificar la fuente de error dista mucho de las medidas anteriores y posteriores en latitud y longitud, o por contrario la latitud es la que se ha desplazado mucho, mientras que la longitud está dando un valor coherente.

Un error previamente conocido, y que ha supuesto una mayor problemática para desarrollar este trabajo, es el de la imposibilidad de realizar medidas durante los equinoccios (21 de marzo y 22 de septiembre). Alrededor de estos períodos la duración del día es prácticamente idéntica en la mayoría del planeta, con lo cual no se consigue diferenciar latitudes a partir de éste parámetro.

### 1.3.2 Campo Magnético Terrestre

Debido a la teórica precisión de 150 km en latitud en el mejor caso por parte de la geolocalización por luz ambiente, y la imposibilidad de realizar medidas en ciertos períodos del año, así como proteger ante los otros posibles errores mencionados en el apartado 1.3.1, se planteó utilizar la geolocalización por campo magnético terrestre como una mejora del algoritmo, (ya que el campo magnético no varía según la fecha y no comparte la mayoría de fuentes de error con la geolocalización por luz).

El campo electromagnético a lo largo de la tierra se obtiene mediante medidas satélite obtenidas a partir de modelos de campo [18]. A partir del campo electromagnético medido, se obtiene el campo magnético terrestre. Este campo se puede describir con elementos de referencia como el norte magnético, el norte geográfico, etc. Y a partir de éstos se obtiene la declinación y la inclinación magnética.

La inclinación magnética [19] es el ángulo que forma el vector de campo en la posición donde se mide respecto al vector de campo horizontal.

Es de 90° en el polo norte magnético, va disminuyendo cuando bajamos en la esfera terrestre, formando 0° en el ecuador, y sigue rotando hasta llegar a -90° en el polo sur. A pesar de esto, no es regular.

Se puede ver claramente esta explicación en la figura 5.a, en la que se representa con curvas de nivel la inclinación magnética en cada localización del planeta.

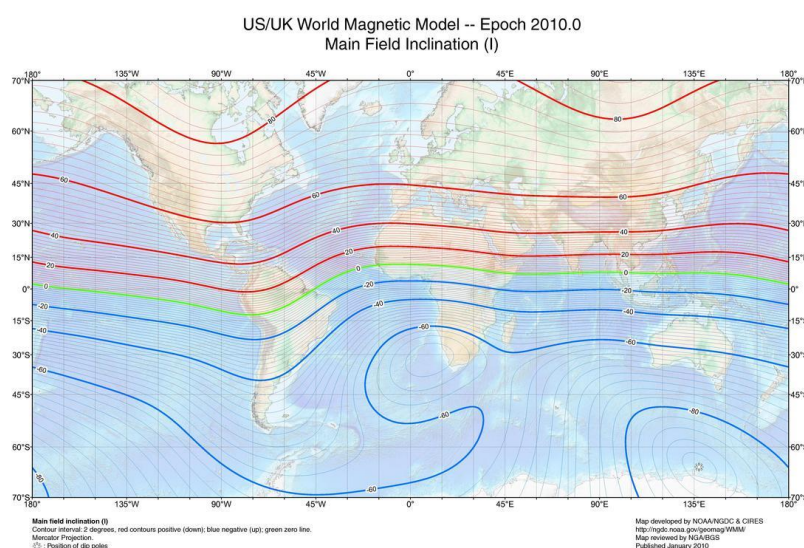


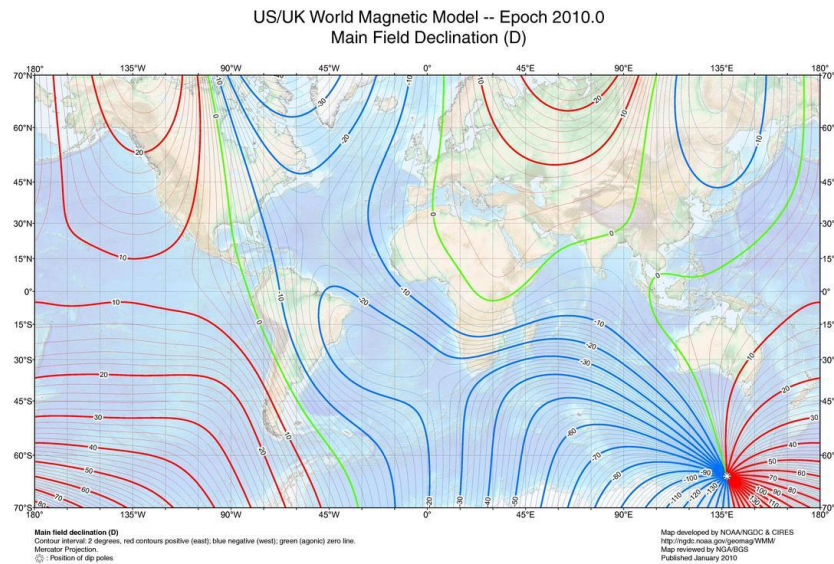
Figura 5.a - Inclinación magnética

Así pues, obteniendo la inclinación del campo magnético en la localización del observador se obtendría el primer trazo para obtener el punto, es decir, las coordenadas de éste.

La declinación magnética [20] es el ángulo que forma el norte magnético con el norte geográfico en la posición del observador.

Este parámetro varía en función de la posición de dicho observador, y es crítico cuando se requiere orientación con una brújula, ya que dependiendo de la zona del planeta en que se mida, puede inducir a un error de 30°, 50° o 70° a la hora de obtener los puntos cardinales con brújula, o incluso cambiar de +10° a -130° en unos cuantos kilómetros.

Se pueden observar estos cambios en la figura 5.b, en la que se representa con curvas de nivel la declinación en cada punto del planeta.



**Figura 5.b - Declinación magnética**

Así pues, con la inclinación y la declinación en el punto del observador se podría obtener un punto de intersección entre las curvas de nivel de ambos parámetros, obteniendo así la localización del observador.

Esto sería el caso ideal, pero no es así en la realidad, ya que una de estas líneas (dejando aparte el hecho de que no es un sistema muy preciso) podría cortar en más de un punto con la otra, dando así varias posibles localizaciones para esos niveles de inclinación y declinación.

Se necesitaría un tercer parámetro para la triangulación. Este parámetro no es otro que la intensidad de campo magnético.

La intensidad de campo magnético [21] no es constante. Varía en función de la localización en la que se mida, lo cual en principio es prometedor para obtener coordenadas a partir de esta. En la figura 5.c se observan los distintos valores de intensidad de campo magnético alrededor del mundo.

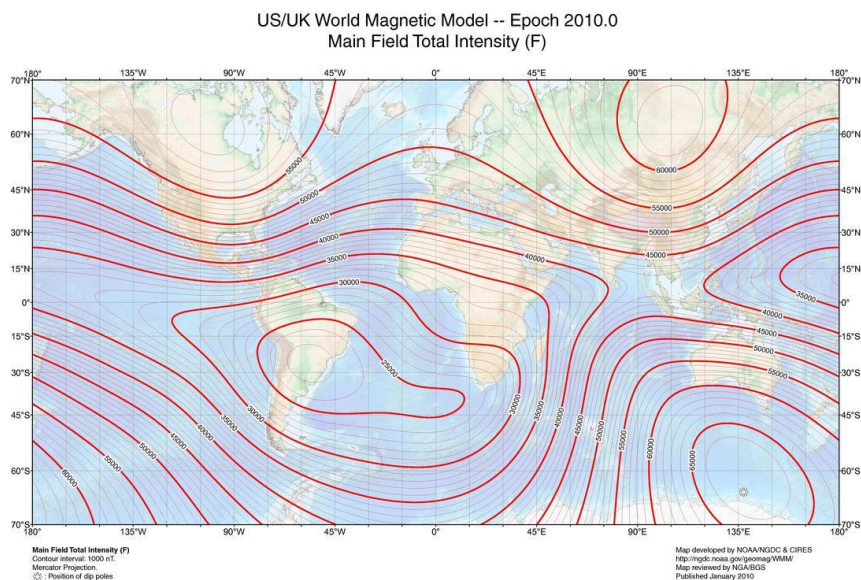


Figura 5.c - Intensidad de campo magnético

Con estas tres componentes se podría realizar una triangulación para obtener unas coordenadas aproximadas para la localización del sensor en el momento de la medida.

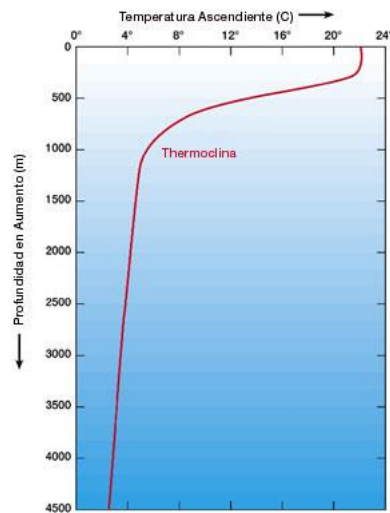
No obstante, debido a que estos parámetros no siguen modelos matemáticos (como es el caso de la geolocalización por luz ambiente), sino que solo se pueden conocer de manera empírica, se necesitaría de un dataset con miles de medidas de campo magnético en distintas localizaciones del planeta. Como no se dispone de dicho dataset y resulta imposible realizarlo personalmente por la falta de recursos para realizar las medidas y poder medir por todo el mundo, y de tiempo para llevar a cabo estas medidas, no se ha podido implementar.

### 1.3.3 Temperatura de superficie marina

Otra posible implementación sería la geolocalización por temperatura superficial del mar, en el caso de medidas en animales marinos. Este sistema como su nombre indica se basaría en tendencias de temperatura marina en su superficie, por tanto se debería realizar un ajuste según la profundidad a la que se encuentre el animal.

Como el TAG desarrollado mide temperatura y profundidad, se podría realizar dicho ajuste de forma muy sencilla y rápida sobre el algoritmo desarrollado.

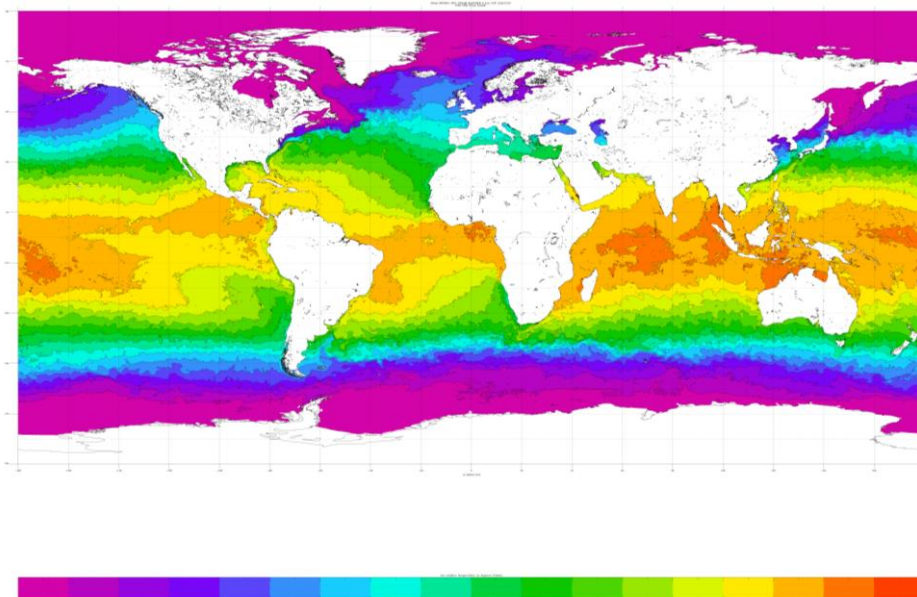




**Figura 6.a - Cambio de la temperatura en función de la profundidad [22]**

Como se puede observar en la figura 6.a, la temperatura sigue una tendencia descendente en función de la profundidad que podría calcularse. Por tanto se podría realizar un ajuste de la medida de temperatura para la profundidad, obteniendo así la estimación de la temperatura en la superficie.

Una vez obtenida la temperatura de la superficie marina, se podría calcular una coordenada aproximada a partir de ésta, ya que según la localización en la que se encuentre existen distintas temperaturas, como se observa en la figura 6.b.



**Figura 6.b - Mapa de temperatura de superficie del mar [23]**

La razón por la que no se ha implementado este sistema en el proyecto, es la misma razón por la que no se implementó la geolocalización por campo magnético, y es la falta de datasets con la temperatura de la superficie del mar en las distintas localizaciones y la imposibilidad de realizarlos personalmente. En caso de disponer de dichos datos se podrían añadir fácilmente estos sistemas al algoritmo, y por consiguiente solucionar la problemática de la falta de medidas en los equinoccios, así como mejorar la precisión del algoritmo obteniendo más medidas, evitando coordenadas inverosímiles causadas por errores en la geolocalización por intensidad de luz solar.

## 1.4 Solución al objetivo

Así pues, debido a estos factores, se decide realizar la investigación de un sistema que sea capaz de obtener localizaciones a partir de unos datos de niveles de luz, sin implementar la solución del campo magnético ni la temperatura de superficie marina, al menos por el momento.

Este sistema por niveles de luz se intentará mejorar en cuanto a precisión en la medida de lo posible.

## 1.5 Estructura del trabajo

En el apartado 2 se explicarán los planteamientos que se le dieron al desarrollo de este algoritmo de geolocalización sobre niveles de intensidad de luz ambiente y los motivos que llevaron a elegir el software GeoLight para llevar a cabo el proyecto.

En el apartado 3, se profundizará en el funcionamiento del algoritmo desarrollado sobre el software GeoLight.

Seguidamente, en el apartado 4 se comentarán los resultados que se han obtenido una vez implementado el algoritmo.

Para finalizar, en el apartado 5 se presentarán las conclusiones, el trabajo futuro de este trabajo, y la valoración personal del alumno.

# 2. Desarrollo

## 2.1 Pasos previos

### 2.1.1 Contextualización del proyecto

En un primer momento se investiga el estado de desarrollo de la geolocalización por luz ambiente en el momento actual.

Inicialmente se estudian una serie de tesis doctorales y artículos científicos que hablan del tema, siendo especialmente interesante el artículo científico "Localization using Magnetometers and Light Sensors [2]", tesis doctoral de Niklas Wählstrom, investigador del departamento de ingeniería eléctrica de la universidad de Linköping, en Suecia.

En este documento se recogen las bases teóricas para el cálculo de las coordenadas a partir de los niveles de luz (resumidas en el apartado 1.1).

También se vieron distintos modelos comerciales de geolocalizadores por sensores de luz, que incluían cierto software para el tratamiento de los datos con el cual sacar las coordenadas, entre los que destaca el modelo "SeaTag" de la empresa Desert Star [24], siendo la tecnología más avanzada del mundo en cuanto al cálculo de coordenadas a partir de la intensidad de la radiación solar.

Este aparato es capaz de reducir el error medio a 40km en longitud y entre 70 km y 130 km en latitud, cifras bastante impresionantes en el contexto de la geolocalización por luz.

Así pues, se llega a la conclusión de que ya existen ciertos aparatos capaces de localizar utilizando éste sistema de geolocalización, y que actualmente se sigue investigando y está en expansión.

### 2.1.2 Planteamiento

Inicialmente se plantea el proyecto como una "construcción" desde cero.

En el GSyM del IGIC, previamente a la realización de éste proyecto y al planteamiento de las posibles soluciones, se desarrolla un dispositivo de medida o TAG, el cual tiene distintos sensores midiendo distintos parámetros del entorno. En la figura 7 podemos ver 3 TAGs midiendo bajo el agua en una piscina con un flotador instalado.

Entre estos parámetros medidos por el TAG se encuentran la radiación solar, temperatura ambiente, campo magnético con sus distintas componentes, profundidad y algunos parámetros auxiliares como la orientación del dispositivo o indicadores que

sirven para señalar si en la medida el dispositivo se encontraba bajo el agua, tuvo un movimiento demasiado brusco que pudiera conllevar un error de medida, u otros.



**Figura 7 - Tag desarrollado y utilizado para las medidas**

Obviamente, cada medida lleva una fecha y una hora correspondiente a la que se realizó. Las medidas se toman con un lapso de 1 minuto. Este sensor servirá para realizar pruebas empíricas del funcionamiento del sistema, así como prototipo para futuros TAGs que se colocarían en los animales.

Una vez desarrollado el hardware quedaba la etapa de software para el procesado de los datos de las medidas realizadas. Como hemos dicho anteriormente, inicialmente se planteó un proyecto de construcción desde cero, es decir que el software se programaría desde cero a partir de las ecuaciones y cálculos astronómicos teóricos.

Así pues se inicia otra etapa de investigación en la que se empieza a estudiar detalladamente los principios teóricos fundamentales para el cálculo de coordenadas a partir de la intensidad de luz ambiente. Además de la tesis de la referencia [2] se estudian otros artículos científicos como “Geolocation by Light Level [25]”, o “Geolocation by light accuracy and precision affected by environmental factors [17]”.

De la misma forma, debido a que en dichos documentos no se profundiza en el funcionamiento a nivel fundamental del algoritmo, sino que más bien se destacan las virtudes y los problemas que enfrenta el sistema, se investiga de forma externa a los documentos todos los principios que se mencionan (ecuaciones astronómicas, cálculos y transformaciones entre sistemas de coordenadas, etc.) con información sacada de internet. Las referencias a los distintos documentos consultados se encuentran en el apartado 6.

### 2.1.3 Cambio de perspectiva: GeoLight

Mientras se investigaba desde el punto de vista de construcción desde cero a partir de los principios teóricos fundamentales, se encuentra una posible solución que podría ahorrar muchísimo tiempo en la asimilación de los conocimientos básicos teóricos, así como en la construcción del algoritmo para calcular las coordenadas.

Ésta solución lleva como nombre “GeoLight” [1]. Se trata de una librería dedicada específicamente a la geolocalización por sensores de luz, así como la importación y exportación de los datos a distintos formatos. Dicha librería está implementada en el lenguaje de programación R, un lenguaje matemático orientado a análisis estadístico. Se trata de un software de código abierto y totalmente gratuito, lo cual es útil a la hora de investigar el código, realizar modificaciones e incluso migrar el algoritmo o parte de éste a otro lenguaje de programación.

Así pues se decide orientar el proyecto al desarrollo con GeoLight, al menos por el momento, hasta ver la validez de su algoritmo.

## 2.2 Set-up R y GeoLight

Se empieza buscando información en la página web oficial del lenguaje R.

Como se reiterará más adelante, la información de ayuda y tutoriales acerca de este lenguaje de programación es más bien escasa, confusa e incompleta, así que pasan varias semanas hasta conseguir tener el entorno, el software de R, y la librería de GeoLight instalados y listos para ser utilizados.

Inicialmente se instala el software R, que es un sistema en tiempo de ejecución como es el Java Runtime Environment. Posteriormente se instalan una serie de requerimientos para el correcto funcionamiento de éste.

Debido a la previa experiencia con el entorno de programación *Eclipse*, se decide utilizar el plugin para programar en R que tiene éste implementado. Tiempo después se ve que ésta solución es inviable debido a las complicaciones, errores, excepciones, y dificultades que supone trabajar con R en un entorno que no es el propio.

Así pues, se decide utilizar el entorno oficial de R, el RStudio. Inicialmente presenta muchas dificultades para instalar el paquete GeoLight, ya que éste tiene 7 paquetes como dependencias, cada uno de los cuales tiene otros tantos paquetes como dependencias. Se acaba por instalar todas las 46 dependencias por orden jerárquico, algunas de las cuales tienen ciertas particularidades para instalarlas que no se ven reflejadas en ningún tutorial ni información de ayuda en la web oficial de R.

Finalmente, tras varias semanas se consigue tener listo el RStudio funcionando con R y el paquete GeoLight compilado y listo para ser utilizado.

Debido a la gran complejidad que supone la instalación del software necesario y la compilación de los paquetes para tener el GeoLight listo y en funcionamiento, a continuación se expone un manual paso a paso de cómo instalar R, RStudio y GeoLight desde cero, con enlaces de instalación incluidos.

## 2.2.1 Instrucciones de instalación de GeoLight

Los archivos adjuntos a este manual son para windows y son totalmente utilizables, pero se dejan los enlaces para descargar el software necesario para otros sistemas operativos, así como para buscar versiones más recientes.

### 1. Instalación del software R

Instalar R ejecutando el archivo "R-3.1.2-win". La última versión se puede encontrar en:

<http://cran.es.r-project.org/>

Aquí se elige el sistema operativo y se descarga en "install R for the first time".

### 2. Instalación del entorno

Instalar Rstudio ejecutando el archivo "RStudio-0.98.1102". La última versión se puede encontrar en:

<http://www.rstudio.com/products/rstudio/download/>

Simplemente se hace click en el archivo correspondiente al sistema operativo deseado.

### 3. Instalación del paquete de utilidades

Instalar Rtools ejecutando el archivo "Rtools32". La última versión se puede encontrar en:

<http://cran.es.r-project.org/>

Es el mismo sitio donde se encuentra el software de R. En lugar de hacer clic en "install R for the first time", clicar en "Rtools" a la izquierda de la página, y elegir la última versión de la lista que aparece arriba del todo (en la cual la columna "Frozen" indica "NO").

## 4. Instalación dependencias GeoLight

Instalar las dependencias del paquete GeoLight, pero NO el propio GeoLight. Estas dependencias están en la carpeta anexa "R Packages" ordenadas jerárquicamente por orden de dependencia. Cuando se instale un paquete, en caso de no tener todas las dependencias necesarias instaladas, surgirá un error, el cual indicará las dependencias que hace falta instalar previamente.

Para buscar versiones más recientes de los paquetes y sus dependencias, simplemente buscar en google "CRAN [nombre del paquete]", y el primer resultado suele ser el apropiado.

### Como instalar las dependencias (o cualquier paquete)

Entrar en Rstudio, y en el menú de "Tools" seleccionar "Install Packages", y en "Install from" seleccionar "Package Archive File". Seleccionar el paquete a instalar, y esperar un "DONE" en la consola, que indicará que este paquete se ha instalado satisfactoriamente.

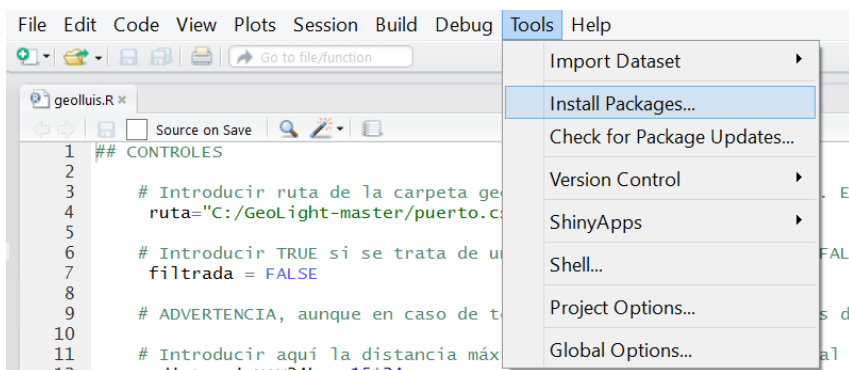


Figura 8.a - Menú de instalación de paquetes

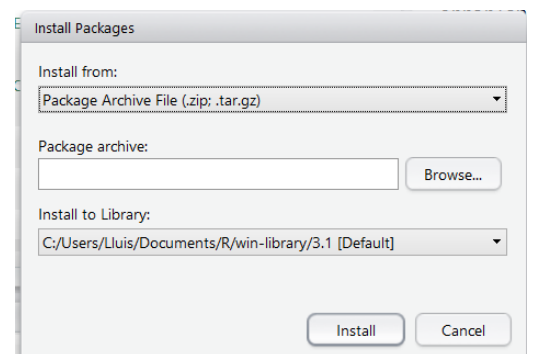


Figura 8.b - Opciones de instalación de paquetes

```

Console C:/GeoLight-master/
>
>
>
>
>
>
> install.packages("E:/Downloads/zoo_1.7-11.tar.gz", repos = NULL, type = "source")
Installing package into 'C:/Users/Lluis/Documents/R/win-library/3.1'
(as 'lib' is unspecified)
* installing *source* package 'zoo' ...
** package 'zoo' successfully unpacked and MD5 sums checked
** libs

*** arch - i386
gcc -m32 -I"C:/PROGRA~1/R/R-31~1.2/include" -DNDEBUG -I"d:/RCompile/CRANpkg/extralibs64/local/inclu
de" -O3 -Wall -std=gnu99 -mtune=core2 -c coredata.c -o coredata.o
gcc -m32 -I"C:/PROGRA~1/R/R-31~1.2/include" -DNDEBUG -I"d:/RCompile/CRANpkg/extralibs64/local/inclu
de" -O3 -Wall -std=gnu99 -mtune=core2 -c init.c -o init.o
gcc -m32 -I"C:/PROGRA~1/R/R-31~1.2/include" -DNDEBUG -I"d:/RCompile/CRANpkg/extralibs64/local/inclu
de" -O3 -Wall -std=gnu99 -mtune=core2 -c lag.c -o lag.o
gcc -m32 -shared -s -static-libgcc -o zoo.dll tmp.def coredata.o init.o lag.o -Ld:/RCompile/CRANpkg/ext
ralibs64/local/lib/i386 -Ld:/RCompile/CRANpkg/extralibs64/local/lib -LC:/PROGRA~1/R/R-31~1.2/bin/i386 -lR
installing to C:/Users/Lluis/Documents/R/win-library/3.1/zoo/libs/i386

*** arch - x64
gcc -m64 -I"C:/PROGRA~1/R/R-31~1.2/include" -DNDEBUG -I"d:/RCompile/CRANpkg/extralibs64/local/inclu
de" -O2 -Wall -std=gnu99 -mtune=core2 -c coredata.c -o coredata.o
gcc -m64 -I"C:/PROGRA~1/R/R-31~1.2/include" -DNDEBUG -I"d:/RCompile/CRANpkg/extralibs64/local/inclu
de" -O2 -Wall -std=gnu99 -mtune=core2 -c init.c -o init.o
gcc -m64 -I"C:/PROGRA~1/R/R-31~1.2/include" -DNDEBUG -I"d:/RCompile/CRANpkg/extralibs64/local/inclu
de" -O2 -Wall -std=gnu99 -mtune=core2 -c lag.c -o lag.o
gcc -m64 -shared -s -static-libgcc -o zoo.dll tmp.def coredata.o init.o lag.o -Ld:/RCompile/CRANpkg/ext
ralibs64/local/lib/x64 -Ld:/RCompile/CRANpkg/extralibs64/local/lib -LC:/PROGRA~1/R/R-31~1.2/bin/x64 -lR
installing to C:/Users/Lluis/Documents/R/win-library/3.1/zoo/libs/x64
** R
** demo
** inst
** preparing package for lazy loading
** help
*** installing help indices
** building package indices
** installing vignettes
** testing if installed package can be loaded
*** arch - i386
*** arch - x64
* DONE (zoo)

```

Figura 8.c - Resultado de consola en instalación satisfactoria

La única excepción es el paquete Rcurl, que necesita instalarse primero con el archivo de origen, el cual dará un error y no el "DONE" esperado, y después instalar el binario "r-devel", que corregirá este error y dará ya el "DONE".

- Reference manual: [RCurl.pdf](#)
- Package source: [RCurl\\_1.95-4.5.tar.gz](#) ← 1
- Windows binaries: r-devel: [RCurl\\_1.95-4.5.zip](#), r-release: [RCurl\\_1.95-4.5.zip](#), r-oldrel: [RCurl\\_1.95-4.5.zip](#)
- OS X Snow Leopard binaries: r-release: [RCurl\\_1.95-4.5.tgz](#), r-oldrel: [RCurl\\_1.95-4.5.tgz](#)
- OS X Mavericks binaries: r-release: [RCurl\\_1.95-4.5.tgz](#)

Figura 9 - Orden de instalación de las distribuciones del paquete Rcurl

## 5. Instalación y compilación del paquete GeoLight

Crear un proyecto de Rstudio dentro de la carpeta "GeoLight Master", la cual se ha de colocar en la raíz del disco local C. En caso de cambiar la ruta se habrían de cambiar unas cuantas rutas dentro del código del script, pero estas son muy fáciles de encontrar y modificar.



En el menú "Build" del Rstudio, entrar en "Configure Build Tools". Seleccionar la casilla "Generate documentation with Roxygen" y entrar en "Configure" justo a la derecha de esta casilla.

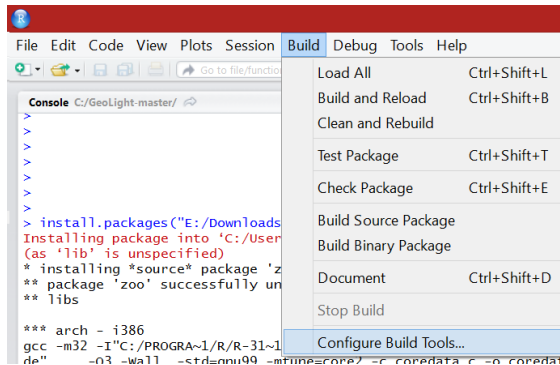


Figura 10.a - Menú de configuración de compilación de paquetes

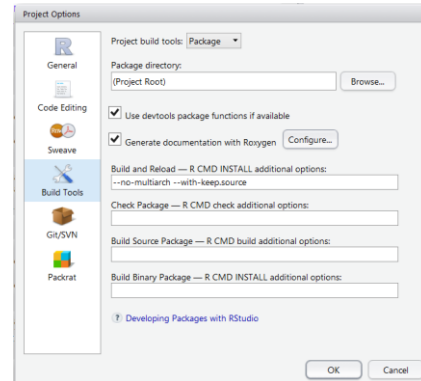


Figura 10.b - Opciones de compilación de paquetes

Seleccionar las casillas "Generate Rd Files" y "Build and Reload".

En el menú "Build" clicar en "Build and Reload", y el paquete GeoLight se instalara y compilara automáticamente.

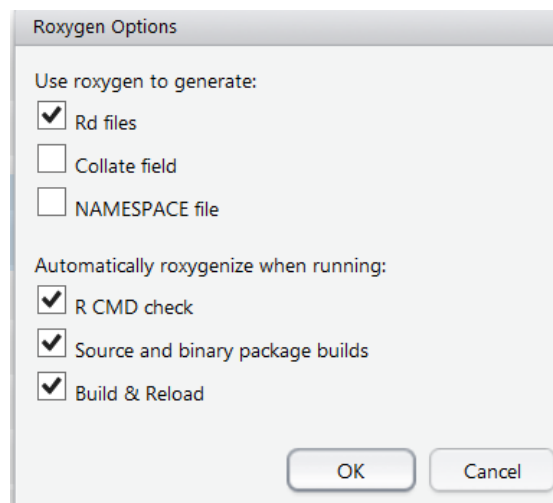


Figura 10.c - Opciones de compilación

**NOTA FINAL IMPORTANTE:** Para utilizar el paquete GeoLight y sus funciones (y obviamente el script desarrollado en este trabajo que se explicará en el apartado 3), cada vez que se abra el Rstudio se habrá de entrar en "file">"open project" y abrir el proyecto creado antes en la carpeta "GeoLight-master". Después de esto, seleccionar la opción "Build and Reload" de antes, o pulsar la combinación de teclas Ctrl+Lshift+B para volver a compilar el paquete y dejarlo a punto para utilizarlo.

## 2.2.2 Utilización del GeoLight

Una vez instaladas todas las herramientas necesarias, tras investigar unos días los comandos básicos de R, los cuales no fueron nada fáciles de deducir (ya que como ya se ha mencionado, la documentación e información de ayuda es pésima, incompleta y escasa), se empiezan a investigar las funciones propias del GeoLight, qué hacen, como se utilizan, y como realizar un script que procese una serie de datos propios y localice las coordenadas donde se realizó la medida utilizándolas.

Se van a exponer las funciones principales del GeoLight que se han utilizado en el Script creado para el procesado de las medidas.

**lightFilter**: Filtro utilizado para eliminar el ruido de los niveles de luz durante la noche (crítico en el momento de detectar el amanecer).

**twilightCalc**: Primera gran función importante del GeoLight. Es la encargada de obtener las horas de los amaneceres y los anocheceres a partir de las fechas-horas con sus correspondientes niveles de luz, a partir de un umbral configurable (por defecto utiliza como umbral 3 unidades arbitrarias, en nuestro caso 3 LUX).

**getElevation**: Segunda gran función del GeoLight. Obtiene el ángulo de elevación adecuado del sol a la hora del amanecer para cálculos posteriores. Una mala calibración de este ángulo supondría un desplazamiento en latitud de todos los puntos obtenidos.

**coord**: Función principal del GeoLight. Se encarga de obtener las coordenadas a partir de las fechas-horas de los amaneceres y atardeceres mediante una serie de cálculos astronómicos. Ésta función y sus cálculos se han explicado con detalle en el apartado 1.1.

**distanceFilter**: Se encarga de identificar coordenadas inverosímiles por contexto. Se le introduce la distancia máxima en kilómetros que el animal u objeto que lleva el TAG es capaz de recorrer en 24 horas. En caso de que se obtuvieran unas coordenadas que distaran más de esta cantidad de kilómetros máxima de la coordenada anterior, se detectaría un error en esa coordenada, pudiéndose utilizar como flag para un ajuste en ésta.

**tripMap**: Obtiene un plot en el que dibuja los puntos obtenidos con coord en un mapa. Este plot se puede exportar con buena calidad a formato JPG y PDF. Una alternativa a este plot sería exportar los datos de las coordenadas en un formato concreto para una herramienta de conversión a formato kml para google earth, y después de convertirlo obtener los puntos sobre un mapa 3D en google earth. En este trabajó se ha utilizado principalmente el plot por defecto del GeoLight, aunque también se ha realizado un ejemplo de plot en Google Earth adjunto en el anexo.

# 3 Script Geolluis.R

## 3.1 Construcción del script

El grueso de este proyecto ha sido la construcción de un script, llamado geolluis.R, que contiene una secuencia de funciones propias, así como un entramado de llamadas a funciones de GeoLight que se complementan con dichas funciones propias. El resultado final de este script son 300 líneas de código en las que se le introduce la ruta del archivo con los datos del TAG, y se obtiene como resultado las coordenadas obtenidas a partir de estos datos.

Se va a proceder a profundizar en el código utilizado en el script geolluis.R. Se mostrará el código tal cual está en el propio script, con los comentarios explicativos situados después del símbolo “#” y en color verde.

### 3.1.1 Código del Script y explicaciones pertinentes

```
# Leer datos del .csv separado por comas en la “ruta” indicada

datos<-read.csv(ruta, header=T)

attach(datos)

# Pasar la columna de fecha-hora de tipo factor a tipo carácter, para poder ser leído
#después

datos[1] <- data.frame(lapply(datos[1], as.character), stringsAsFactors=FALSE)

# Se filtra la luz para reducir el ruido de los niveles de luz durante la noche, lo cual es
#crítico a la hora de realizar el cálculo del amanecer

luzbuena=lightFilter(datos[,2], baseline = NULL, iter = 3)

# Cambiar valores negativos por 0 en caso de tratarse de una señal filtrada debido a
#unos niveles de luz demasiado abruptos.

if(filtrada==TRUE){
  for (i in 1:(length(datos[,2])) ) {
    if(datos[i,2]<0){
      datos[i,2]=0;
    }#endif }#for}#endif
```

```
# -----  
  
# Leer los datos de fecha-hora y luz, y crear una matriz con la fecha y la hora de los  
#distintos amaneceres y atardeceres del dataset. Detección de éstos con umbral.  
  
twilights<-twilightCalc(datos[,1], luzbuena, LightThreshold = TRUE, preSelection =  
TRUE,maxLight = NULL, ask = FALSE, nsee = 500, allTwilights = FALSE)  
  
# -----  
  
# Exportar esta matriz de amaneceres y atardeceres a un .csv separado por comas  
  
write.table(twilights, "C:/GeoLight-master/twilights.csv", sep=",")  
  
# Volver a leer esta matriz desde el .csv exportado y pasarla a formato "character"  
para una correcta lectura posterior por parte de la siguiente función  
  
twilightsGOOD<-read.csv("C:/GeoLight-master/twilights.csv", header=T)  
  
attach(twilightsGOOD)  
  
twilightsGOOD[1] <- data.frame(lapply(twilightsGOOD[1], as.character),  
stringsAsFactors=FALSE)  
  
twilightsGOOD[2] <- data.frame(lapply(twilightsGOOD[2], as.character),  
stringsAsFactors=FALSE)  
  
# Separar las fechas-horas de amaneceres y atardeceres, dentro de la matriz twilights  
  
sunrises<-c()  
  
sunsets<-c()  
  
for (i in 1:(length(twilights[,1])/2) ) {  
  if (twilights[1,3]==1) {  
    sunrises <- c(sunrises, twilightsGOOD[(2*i)-1,1]);  
    sunsets <- c(sunsets, twilightsGOOD[(2*i)-1,2]);  
  } else {  
    sunrises <- c(sunrises, twilightsGOOD[(2*i)-1,2]);  
    sunsets <- c(sunsets, twilightsGOOD[(2*i)-1,1]);  
  } }  
}
```

---

```

# -----

## CALIBRACIÓN Y CÁLCULO DE COORDENADAS

# Cuando se realicen las medidas con el TAG, se debe realizar antes una medida de
#un día y una noche para calibrar. Este día y noche debe localizarse dentro de la tabla
#de medidas (por ejemplo que sea la primera medida dentro de ésta), y se deben
#saber las coordenadas de la localización donde se ha efectuado dicha calibración.

# Se le introduce a esta función la muestra de calibración, así como las coordenadas
#donde se ha efectuado ésta.

elevacion_correcta<-
getElevation(sunrises[numero_muestra_calib],sunsets[numero_muestra_calib],1,c(lon
gitud,latitud),plot=TRUE)

coordenadas<-coord(sunrises, sunsets, 1, degElevation = elevacion_correcta, site =
FALSE, site,note = FALSE)

# CÁLCULO DEL ERROR

#En las siguientes líneas de código se calcula el error cometido en las medidas. Éste
#algoritmo para el cálculo de error está pensado para los datasets que se han obtenido
#con el TAG, los cuales estaban fijos en una posición conocida. Está pensado
#únicamente para comprobar el funcionamiento del software localizador.

#En caso de tener un dataset con localizaciones desconocidas que se quieren calcular,
#no se podría calcular el error en todas las muestras a no ser que se tuviera otro
#fichero de referencia con las coordenadas en cada día, pero se podría rediseñar este
#algoritmo para calcular el error sólomente en las muestras de calibración, de las
#cuales sí que se debería saber la localización.

#Éste algoritmo obtiene el error de localización en grados y lo transforma a kilómetros.

#Error en Longitud

errorlong = c();

for (i in 1:(length(coordenadas[,1])) ) {          errorlong [i] = (longitud -
coordenadas[i,1])*110; }

for (i in 1:(length(errorlong) )) { if(is.na(errorlong[i])){ errorlong[i]=errorlong[i+1];}}

```

### # Error en Latitud

```
errorlat = c();  
  
for (i in 1:(length(coordenadas[,2])) ) {errorlat [i] = (latitud - coordenadas[i,2])*110;}  
  
for (i in 1:(length(errorlat)) ) {if(is.na(errorlat[i])){ errorlat[i]=errorlat[i+1]; } }
```

### # AUTOCALIBRACIÓN

#Éste apartado está pensado para re-calcular la calibración del ángulo de elevación  
#del sol, las coordenadas, y las posteriores tareas en caso de que en la muestra de  
#calibración se obtuviera un error mayor a 50 kilómetros. En el caso ideal en dicha  
#muestra se debería tener un error muy pequeño, ya que el ángulo de elevación se ha  
#calibrado con esa muestra. Así pues, en caso de obtener un error demasiado grande  
#en esta muestra se realizaría un barrido de posibles ángulos de elevación (por tanteo)  
#volviendo a calcular las coordenadas, hasta obtener un error inferior.

```
if(abs(errorlat[numero_muestra_calib])>=50){  
  
elevacion_tanteo=1.3;  
  
while ( abs(errorlat[numero_muestra_calib])>=50 ) {
```

### # Calcular latitud y longitud a partir de estos

```
elevacion_tanteo=elevacion_tanteo+0.1;  
  
coordenadas<-coord(sunrises, sunsets, 1, degElevation = elevacion_tanteo, site =  
FALSE, site,note = FALSE)
```

### # Calculo error

```
errorlong = c();  
  
for (i in 1:(length(coordenadas[,1])) ) { errorlong [i] = (longitud -  
coordenadas[i,1])*110; }  
  
for (i in 1:(length(errorlong)) ) { if(is.na(errorlong[i])){ errorlong[i]=errorlong[i+1];}}  
  
errorlat = c();  
  
for (i in 1:(length(coordenadas[,2])) ) {errorlat [i] = (latitud - coordenadas[i,2])*110;}  
  
for (i in 1:(length(errorlat)) ) {if(is.na(errorlat[i])){ errorlat[i]=errorlat[i+1]; } }  
  
} #while } #if
```

---

```

# -----

#FILTRADO DE DISTANCIA MÁXIMA

#Se filtran los resultados de coordenadas inverosímiles, que distan más que la
#distancia máxima que se puede recorrer en 24 horas de la muestra anterior

if(filtrardistanciamaxima==TRUE){

#Filtrado. Devuelve un vector booleano con TRUE en caso de una coordenada correcta
#y FALSE en caso de una coordenada demasiado distante de la anterior.

  DF <- distanceFilter(sunrises, sunsets, 1, degElevation = elevacion_correcta,
distanciaMAX24h,units = "hour")

  # Para las coordenadas inverosímeles (el animal recorre más distancia de la posible
#en 24h) se cambia esa coordenada por la anterior.

  for (i in 1:length(DF)){

    if (DF[i]==FALSE) {

      coordenadas[i,1]=coordenadas[i-1,1]

      coordenadas[i,2]=coordenadas[i-1,2] }

  } }

#Se plotea el patrón migratorio del animal en un mapa exportable a JPG y PDF

  tripMap(coordenadas, equinox =
TRUE,xlim=c(longitudmin,longitudmax),ylim=c(latitudmin,latitudmax), col = "tomato2",
main = NULL, legend = TRUE, add = FALSE)

#Se exportan las coordenadas a un .csv por si se quieren convertir a otros formatos
#como .kml para Google Earth.

write.table(coordenadas, "C:/GeoLight-master/coordenadas.csv", sep=",")

# Se muestran los resultados del error (medio e individual de cada medida) cometido
#en las medidas

  errormiglong=mean(abs(errorlong)); maxerrorlong=max(abs(errorlong));

  errormiglat=mean(abs(errorlat)); maxerrorlat=max(abs(errorlat));

# end

```

## 3.2 Utilización del script

Al inicio del script se han creado unas cuantas líneas de código que servirán de “controles” para futuros investigadores que quieran utilizar el software GeoLight sin tener que entrar en profundidad en el funcionamiento y los requerimientos específicos de formato que necesita ésta librería. A continuación se explican los pasos previos a la utilización del script, ya que se requiere preparar los datos en un formato específico antes de utilizarlos en éste. Posteriormente se explicaran los “controles” anteriormente citados.

### FORMATO

Los datos de entrada deben estar en formato .csv separado por comas y la primera columna debe ser la de fecha y hora, con el siguiente formato:

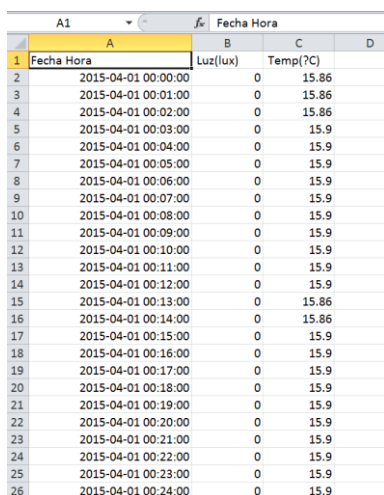
Año-mes-día hora:minuto:segundo

yyyy-mm-dd hh:mm:ss (nomenclatura para formato de celda en excel)

Respetando el número de dígitos para cada unidad, aunque sean 0s.

En la segunda columna deben estar los niveles de luz captados por el sensor a cada fecha-hora correspondiente a la primera columna.

Se debe utilizar el punto “.” como separador decimal, ya que el .csv estará separado por comas.



Fecha Hora	Luz(lux)	Temp(°C)
2015-04-01 00:00:00	0	15.86
2015-04-01 00:01:00	0	15.86
2015-04-01 00:02:00	0	15.86
2015-04-01 00:03:00	0	15.9
2015-04-01 00:04:00	0	15.9
2015-04-01 00:05:00	0	15.9
2015-04-01 00:06:00	0	15.9
2015-04-01 00:07:00	0	15.9
2015-04-01 00:08:00	0	15.9
2015-04-01 00:09:00	0	15.9
2015-04-01 00:10:00	0	15.9
2015-04-01 00:11:00	0	15.9
2015-04-01 00:12:00	0	15.9
2015-04-01 00:13:00	0	15.86
2015-04-01 00:14:00	0	15.86
2015-04-01 00:15:00	0	15.9
2015-04-01 00:16:00	0	15.9
2015-04-01 00:17:00	0	15.9
2015-04-01 00:18:00	0	15.9
2015-04-01 00:19:00	0	15.9
2015-04-01 00:20:00	0	15.9
2015-04-01 00:21:00	0	15.9
2015-04-01 00:22:00	0	15.9
2015-04-01 00:23:00	0	15.9
2015-04-01 00:24:00	0	15.9

Figura 11.a - Ejemplo de tabla de valores

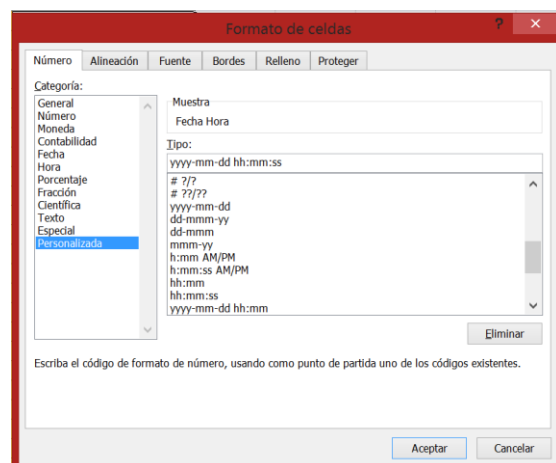


Figura 11.b - Formato de fecha excel



Una vez se tenga el formato adecuado, se debe estar seguro de guardarlo en formato .csv separado por comas.

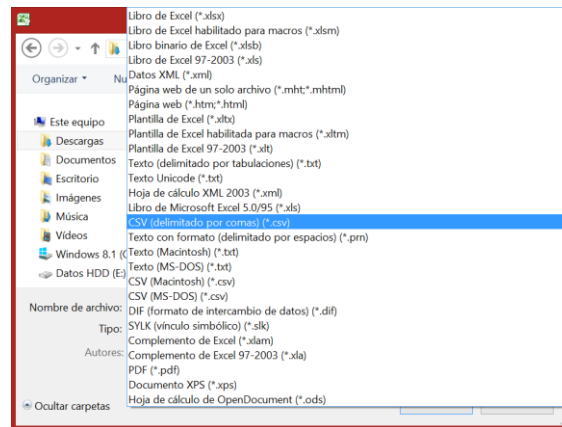


Figura 11.c - Guardado en formato .csv

## CALIBRACIÓN

La primera muestra debe ser a las 00:00, y los intervalos de tiempo entre muestras deben ser totalmente regulares. Además, se han de saber las coordenadas del TAG (el sensor) el primer día, que será el día de calibración.

Se recomienda dejar el TAG inmóvil en esta localización de calibración durante unos días, para asegurarse de que habrá al menos una “buena señal” (con una subida y bajada suave y regular) para calibrar de forma óptima.

En caso de ver que el primer día la señal tiene algún tipo de subida o bajada abrupta o picos que la deformen, eliminar los datos hasta las 00:00 del día siguiente, que pasará a ser el nuevo día de calibración, o especificar en los “controles” del script el número de muestra.

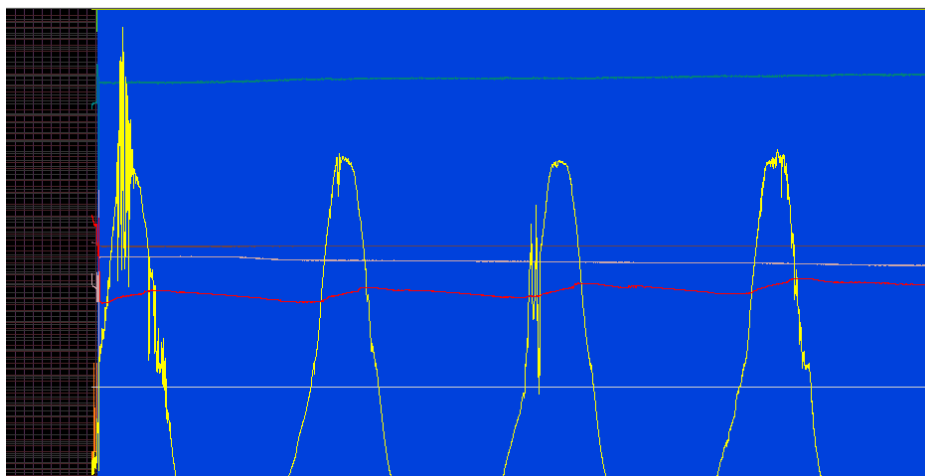


Figura 12 - Valores de luz en un dataset medido con el TAG

Por ejemplo, en la señal de la figura 12, como la primera muestra ha resultado perjudicada por algún tipo de movimiento o dificultad del sensor a la hora de medir, sería conveniente eliminar la primera muestra de calibración o especificar en el script que la muestra que se utilizará como muestra de calibración será la segunda, aparentemente regular y perfecta.

### UTILIZACIÓN DE LOS CONTROLES

Una vez se tenga el .csv con el formato correcto, se abrirá el script “geolluis.R” desde el Rstudio en “file”>”open”.

En la parte superior del script se pueden observar unas líneas de código con los comentarios explicativos donde se encuentran algunas variables, las cuales hacen de “controles” para la utilización del script. Estos controles son:

**Ruta:** La ruta y nombre del archivo .csv con los datos.

Ejemplo: "C:/GeoLight-master/puerto.csv";

**Filtrada:** TRUE o FALSE, indicando si los niveles de luz de la señal se han filtrado para corregir los desniveles y picos demasiado pronunciados que pudieran crear falsos amaneceres o atardeceres. En caso de ser filtrada seguramente se hayan creado algunos valores negativos en las bajadas en los atardeceres, y si no se corrigen éstos introducen gran cantidad de error.



Figura 13 - Ejemplo señal abrupta y filtrada

En la figura 13 observamos en amarillo la señal abrupta, y en rojo la señal filtrada.

**DistanciaMAX24h:** Distancia máxima en (Km) que puede recorrer el animal en 24 horas, para descartar resultados inverosímiles.

**Filtrardistanciamaxima:** TRUE o FALSE, indicando si se quiere utilizar o no esta distancia máxima. Se recomienda probar de las dos formas, para comprobar cuando se obtiene un menor error.

**Calibración:** Coordenadas del día de calibración.

latitud = 39; longitud = -0.16;

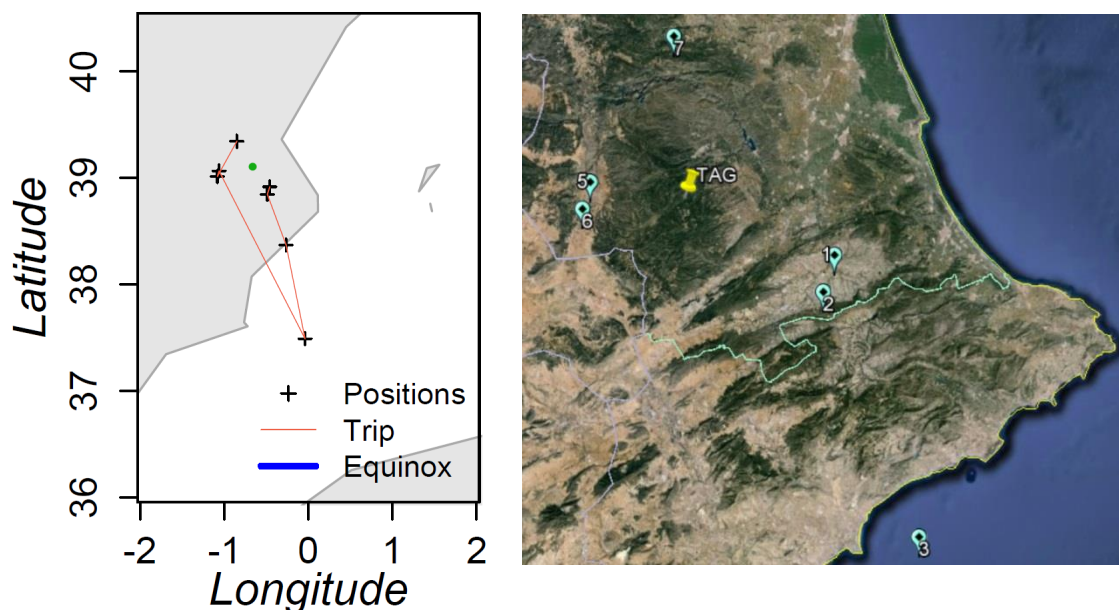
**OPCIONAL: Límites del mapa a plotear.**

**latitudmax=42    longitudmax=2    latitudmin=36    longitudmin=-2**

Una vez se hayan rellenado todos los datos, se ha de seleccionar todo el texto del script y pulsar el botón “run”, en la parte superior de la ventana del script. Una alternativa es simplemente copiar y pegar el código en la consola.

Después de ejecutar el código, si no ha habido ningún error de formato o similares, en la carpeta “GeoLight-master” aparecerán 2 nuevos archivos de tipo .csv, uno llamado “twilights”, siendo una lista de la fecha y hora de los amaneceres y atardeceres, y otro llamado “coordenadas”, con las coordenadas obtenidas en estas fechas-horas.

Este último .csv se puede utilizar para exportar estas coordenadas a formato kml para Google Earth, como se ha explicado anteriormente, aunque en el propio Rstudio aparecerá un plot exportable a jpeg o pdf, con los puntos de las coordenadas obtenidas, dibujadas en un mapa dentro de los límites especificados en los controles. Se pueden ver estos resultados en la figura 14.



**Fig. 14 – Ejemplo de plot en RStudio a la izquierda y exportacion a Google Earth a la derecha**

Estas explicaciones se han realizado para futuros investigadores, para no requerir un período de aprendizaje de los conceptos teóricos y del lenguaje R a la hora de tratar señales mediante el software GeoLight, sino simplemente utilizar los controles del script y obtener resultados.

## 4. Resultados experimentales

A lo largo del desarrollo de este script se ha ido comprobando su funcionamiento con una serie de señales reales. Estas señales han sido medidas a partir del TAG diseñado por el grupo GSyM del IGIC. El TAG se situó repetidas veces en 3 localizaciones distintas.

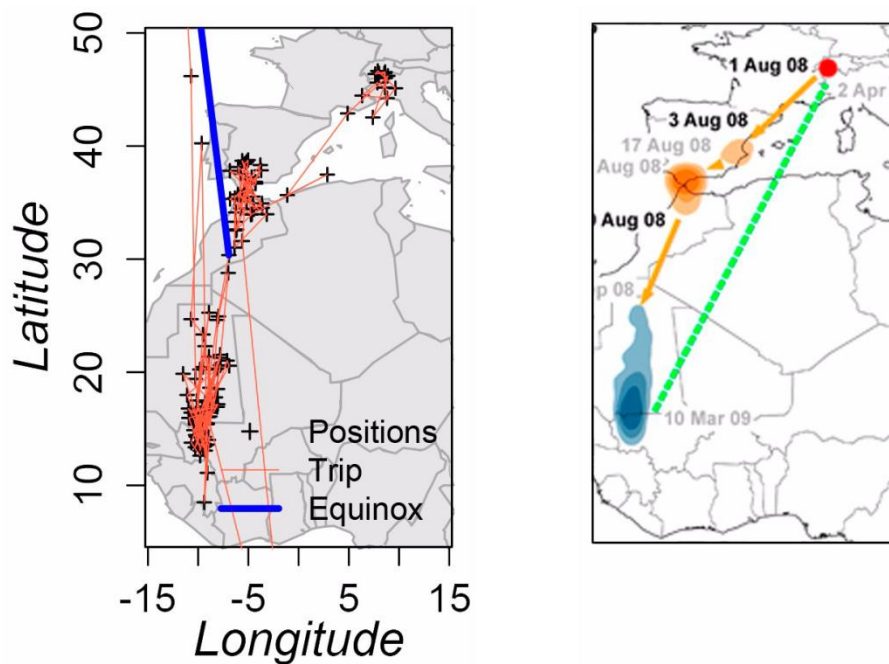
La primera localización fue en el fondo de una piscina, con una profundidad aproximada de 2 metros. Inicialmente no se obtuvieron resultados debido a que el script aún estaba en desarrollo, y todavía se estaba en la etapa de aprendizaje del lenguaje R. Así pues, al principio se obtuvieron muchos errores de compilación debido a errores de formato de datos, de tipo de objeto creado, y de llamada a funciones.

Después de superar una serie de obstáculos y errores debidos a la sintaxis del lenguaje R y ciertas incoherencias internas dentro del GeoLight (como sería que el formato de salida de los datos de una función se obtienen con un objeto de tipo fecha, pero después se tiene que proceder a una complicada transformación a tipo carácter para introducirlo a la siguiente función) que se tradujo en un significativo período de tiempo utilizado para averiguar cómo solucionar estos problemas, se empezaron a ver los resultados de las medidas en la piscina.

Estas medidas presentaban un error excesivamente grande, de cientos o incluso miles de kilómetros en latitud. En longitud en cambio se obtuvieron unos resultados excelentes, de menos de 50 km de error.

Después de numerosos intentos y correcciones del script se validó el correcto funcionamiento de éste con un dataset pre-establecido del GeoLight que contenía datos de luz captados por un sensor que llevó un pájaro, en concreto un *Upupa Epops* (Abubilla o Cuco europeo), a lo largo de su período migratorio.

Se compararon los resultados obtenidos con el patrón migratorio conocido real de éste pájaro, obteniendo unos resultados sorprendentemente buenos. Dicha comparación se puede observar en la figura 15.



**Figura 15 - A la izquierda, los resultados obtenidos. A la derecha, el patrón migratorio teórico.**

De esta forma, se observan unos resultados muy buenos con esta señal, lo que llevó a investigar los posibles motivos de error en las pruebas en la piscina. Se concluyó que este error provenía de dos factores:

El primero, el hecho de no tener visión directa con el sol, ya que el borde de la piscina evitaba una visión del sol justo en el amanecer y el anochecer, dando días notablemente más cortos, y por tanto provocando un gran error en latitud.

El segundo factor, el cual provocó un gran impacto en el proyecto era el equinoccio de primavera. Como se ha comentado en el apartado 1.3.1, una de las limitaciones de este sistema es la incapacidad de dar resultados en fechas cercanas a los equinoccios, ya que las horas de sol se igualan en casi todo el planeta, haciendo que los resultados puedan dar cualquier latitud, y por tanto teniendo miles de kilómetros de error. Justo las primeras medidas se realizaron a inicios de marzo, fechas críticas para el equinoccio.

Este factor provocó un sensible retraso en la comprobación del algoritmo, hasta pasadas unas semanas del equinoccio, tiempo que se dedicó a implementar nuevas funciones y características al script que finalmente tuvieron una gran importancia, como sería el planteado, o el cálculo del error. También se investigaron opciones para evitar este obstáculo en los equinoccios, como la localización por temperatura (apartado 1.3.3).

Después de este obstáculo, se realizó una medida en el segundo emplazamiento, el puerto de Gandia.

Se colocó el TAG bajo el agua a 3 metros de profundidad, teniendo una visión más directa del sol tanto en el amanecer como en el anochecer, dado que no se presentaban bordes que supusieran un obstáculo para captar la luz del sol en los crepúsculos. Se procesó esta señal obteniendo unos resultados excelentes, con un error medio de 112 km en latitud y 21 km en longitud.

Estos resultados se observan en la figura 16.a. El punto verde representa la localización real del TAG, y las cruces los puntos donde el software ha calculado la posición de éste.

Se han obtenido dos puntos más alejados, que coinciden con los días en que la señal presentaba una forma más abrupta por posibles problemas en la medida. Aun así, estos puntos no distan más de 200 km.

Estos resultados son muy positivos, ya que aunque se han realizado en una localización fija, presentan un error similar en latitud, y dos veces menor en longitud comparándolo con el "SeaTag" de Desert Star, que como hemos mencionado en el apartado 2.1.1 es el instrumento más avanzado en este campo.

Durante las siguientes semanas se realizaron una serie de correcciones, se añadieron funciones y características al script (como los algoritmos para obtener y mostrar el error en latitud y longitud en km, o los controles para la utilización futura del script por parte de investigadores que desconozcan el funcionamiento del algoritmo), mientras se realizaba una segunda medida en el tercer emplazamiento, que fue a dos metros sobre el suelo de una azotea en la Poble Llarga.

Se obtuvieron de nuevo unos resultados complicados. El error medio era de 1700 km, pero la mayor parte de este error proviene de las dos primeras muestras, las cuales presentaban unos picos muy abruptos de los que se obtenían resultados inverosímiles. Posteriormente se averiguó que estos tramos de señal correspondían al período entre la inicialización de la medida en el laboratorio de la universidad, y la colocación en su localización (primera muestra), y de días con viento intenso (segunda muestra), que hacían vibrar el dispositivo por no estar fijado correctamente.

Estos resultados se pueden observar en la figura 16.b, que al igual que la anterior figura, tiene el emplazamiento real marcado con un punto verde, y las coordenadas calculadas marcadas con cruces.

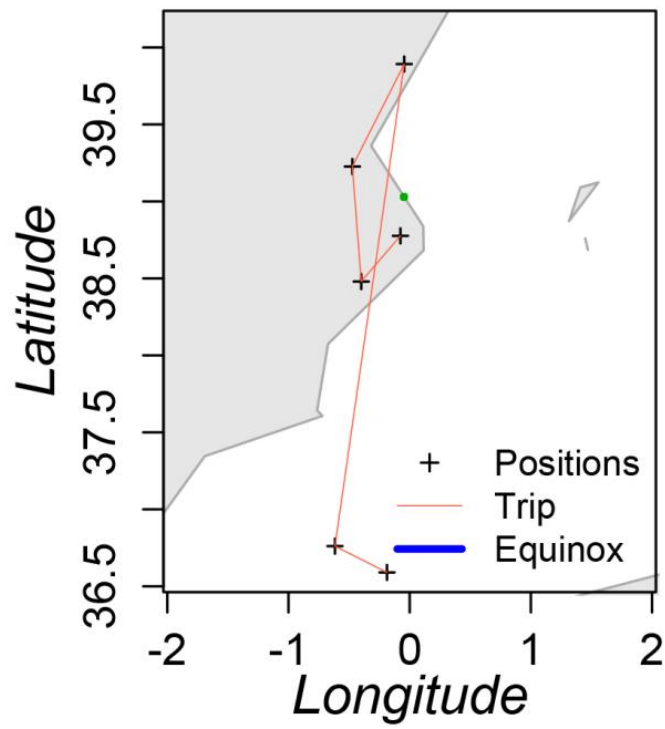


Figura 16.a - Resultados del Tag en el puerto

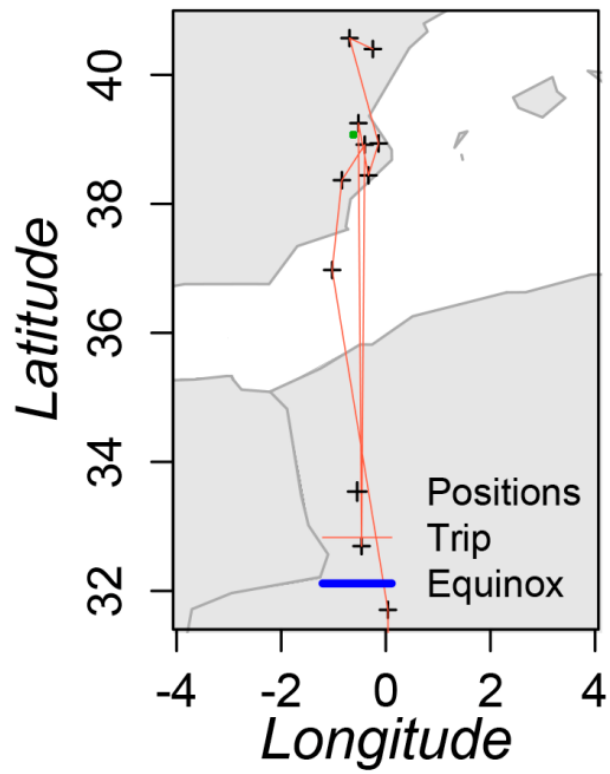


Figura 16.b – Primeros resultados del TAG en azotea

Obviando estas dos medidas que inducen a un error excesivamente grande (se pueden observar en la parte inferior de la figura 16.b) se obtiene un error medio de 140 km en latitud y 40 km en longitud, resultados similares también a los del “SeaTag” de Desert Star.

La última medida que se realizó fue también en la azotea, pero esta vez se fijó el TAG de forma que no pudiera vibrar. Se obtuvieron los mejores resultados hasta la fecha, con mucha diferencia. El error en longitud fue de 53 km en longitud, y de solamente 44 km en latitud, un error excepcionalmente bajo comparado con cualquier sistema de geolocalización por sensores de luz que se conozca.

Se observan resultados muy positivos, obteniendo varias muestras con distancias cercanas e incluso inferiores a los 10 km de error.

Esto demuestra que al tener un dataset con datos de luz medidos de forma ideal (visión directa, sin vibración, y movimientos poco bruscos) junto con una calibración óptima se pueden obtener resultados sorprendentemente precisos. Éstos se pueden observar en la figura 16.c.

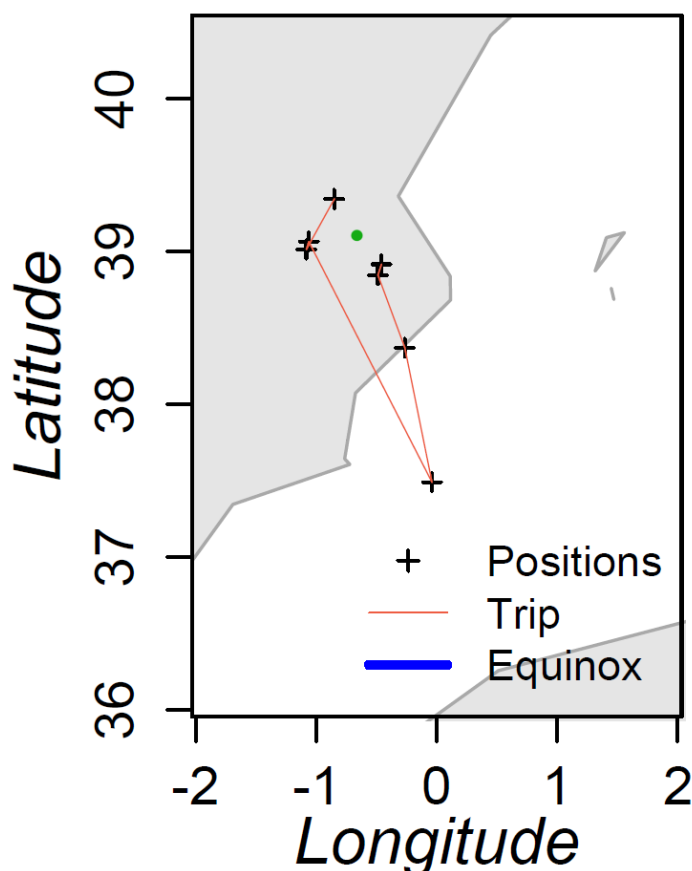


Figura 16.c – Segundos resultados del TAG en azotea



# 5. Conclusiones

Como se especifica en el apartado 1.2, los objetivos marcados para este trabajo son:

- Diseñar un algoritmo que sea capaz de geolocalizar a un atún.
- Implementar el algoritmo en una plataforma software para poder ser utilizado en casos reales.
- Testear el algoritmo con datos registrados de la luz solar y la temperatura del mar.

Se han cumplido estos objetivos, dado que se ha desarrollado un script de código R basado en el software GeoLight capaz de obtener las coordenadas del sensor a partir de los datos de luz captados por éste.

Se ha testado este algoritmo con datos reales obtenidos mediante el TAG desarrollado por el GSyM del IGIC.

Así pues se ha cumplido con el objetivo, y se tiene un sistema capaz de obtener coordenadas con una precisión de 44km en latitud y 53km en longitud.

El script implementado tiene una guía paso a paso, y unos controles relativamente sencillos de utilizar para futuras investigaciones, de forma que no se requiera un profundo conocimiento del software ni de la geolocalización para obtener resultados a partir de niveles de luz.

Como se menciona en el apartado 2.1.3 se ha elegido plantear el proyecto como un desarrollo del software GeoLight debido al atajo que supone no tener que investigar de forma profunda los principios fundamentales de la geolocalización por luz, llegar a desarrollar un algoritmo a partir de estos que sea capaz de localizar a partir de los niveles de luz captados, e implementar dicho algoritmo en un lenguaje de programación adecuado.

En caso de haber tenido que realizar este estudio teórico fundamental y posterior desarrollo, posiblemente sólo se hubiera podido llegar a una etapa temprana del desarrollo del algoritmo, el cual seguramente no tendría todas las posibilidades que tiene el script Geolluis.R, ni su precisión.

Por tanto, el alumno cree que la decisión de darle este enfoque al trabajo fue la idónea para llegar a los objetivos.

## 5.1 Trabajo futuro

Existen muchas mejoras al sistema desarrollado que podrían implementarse en el futuro para aumentar el rendimiento de éste. Entre ellas se encuentran las ya mencionadas, geolocalización por campo magnético terrestre, y por temperatura de superficie marina, que como se menciona en los apartados 1.3.2 y 1.3.3 no se han podido añadir al algoritmo debido a que no siguen modelos matemáticos, y a la falta de conjuntos de datos empíricos utilizables (y la imposibilidad de realizarlos personalmente), pero en caso de disponer de éstos se podrían implementar de forma sencilla y rápida.

Otra posible mejora al algoritmo sería la obtención de puntos adicionales mediante la utilización de las horas de noche, además de las horas de luz solar durante el día.

Se podría modificar el algoritmo de forma que además de obtener una medida tomando las horas de luz solar como parámetro de entrada, también obtuviera otra medida con las horas de noche, y obteniendo una duración del día alternativa a partir de estas.

Una alternativa más simple a modificar el algoritmo sería utilizar el mismo pero añadiendo crepúsculos obtenidos de forma semi-manual. Es decir, en una primera medida se toman la hora del amanecer y la del anochecer, y antes de realizar la medida del siguiente amanecer y anochecer, se obtiene una medida intermedia. En esta medida intermedia se considerará como amanecer la hora en que salió el sol en el segundo día, y como anochecer la hora del anochecer del primer día, pero obviamente sumando 24 horas.

Aumentando el número de puntos tendríamos un mejor seguimiento del animal y un algoritmo más sólido frente a errores.

Otra posible mejora sería la adición al algoritmo del GeoLight de la ecuación del tiempo [2] [26].

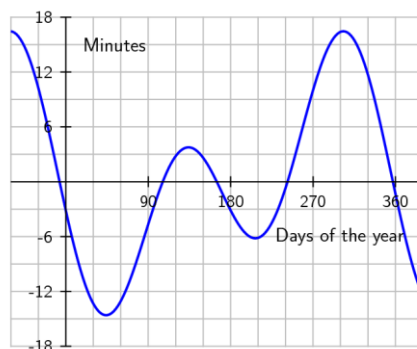


Figura 17 - Variación del aumento y disminución de tiempo de luz solar diaria

En la figura 17 observamos cómo el cambio de tiempo de luz solar diaria a lo largo del año no es regular, sino que dependiendo de la época del año cada día variará de forma distinta cada día (por ejemplo, en enero cada día se tienen 5 minutos menos de luz solar, y en diciembre sólo 1,5). Esto significa que en caso de considerar que las variaciones de tiempo de luz solar diaria son lineales, en un día se podrían haber realizado cálculos con hasta 16 minutos menos de luz solar teórica que la que realmente existe.

Esta variación que depende de la época del año, se obtiene mediante una serie de cálculos sobre las ecuaciones que se observan en la referencia [26]. A partir de estos complicados cálculos se obtiene la llamada ecuación del tiempo. Ésta se observa en la siguiente ecuación en función del número de días desde el último perihelio.

$$M = \frac{2\pi}{t_y} n \quad \text{Ecuación del tiempo en función de } n \text{ [26]}$$

Siendo  $M$  el número de minutos que difiere en ese día del año,  $t_y$  el día del año, y  $n$  el número de días desde el perihelio (punto más cercano al sol).

Como se ha dicho en el apartado 1.1.2, la latitud se calcula mediante la duración del día a partir de la cual el algoritmo realiza una serie de aproximaciones para obtener los parámetros necesarios para calcularla. El GeoLight supone las variaciones de luz solar diaria como lineal, la cual puede no tener mucha importancia en ciertos períodos del año, pero en algunos difiere de la duración del día real hasta 16 minutos.

La implementación de dichas ecuaciones que realizaran este ajuste supondría un aumento significativo en la precisión de las coordenadas calculadas, especialmente en estos períodos del año donde la diferencia de duración del día es tan grande.

### 5.1.1 Aplicaciones

Dicho todo esto, cabría mencionar las posibles aplicaciones que se le podrían dar al sistema desarrollado. Como se ha mencionado a lo largo de toda la memoria, este sistema se puede utilizar para realizar un seguimiento de los patrones migratorios de animales marinos, terrestres o aves.

No obstante, se podría utilizar para otros menesteres, como el seguimiento de barcos u otros vehículos de mercancías, asegurándose que siguen la ruta establecida, pudiendo ver si se han producido desvíos en caso de faltar mercancías.

Las posibilidades son innumerables, basta con que para localizar de forma efectiva sea suficiente una precisión de 50km, ya que prácticamente cualquier ser vivo, dispositivo o máquina que se desee localizar podría llevar el sensor de tan reducidas dimensiones y gran autonomía durante años.

## 5.2 Valoración personal

Este proyecto ha sido un reto para el alumno, ya que ha tenido que desarrollar desde cero un sistema que fuera capaz de conseguir los objetivos marcados, sin ningún tipo de guía de cómo hacerlo, y además realizarlo en un lenguaje desconocido para el alumno y del cual (generalmente) existe tan poco conocimiento y documentación.

El hecho de elegir el desarrollo a partir del software GeoLight ahorró tiempo de investigación y desarrollo, pero aun así el hecho de trabajar a partir de éste ha sido la mayor dificultad encontrada en el trabajo.

El lenguaje de programación R, como ya se ha dicho, es un lenguaje matemático orientado a estadística, lo cual es idóneo para nuestro caso, pero tiene sus particularidades. Estas particularidades, junto con el desconocimiento del lenguaje, la falta de documentación y tutoriales para el aprendizaje de éste, y la mala calidad de los documentos de ayuda de la página oficial del lenguaje R han supuesto sensibles retrasos a la hora de obtener resultados, dado que los errores y excepciones surgidas al poner a prueba el algoritmo fueron el grueso del trabajo y lo más costoso de solucionar.

Aun así, el alumno cree haber solventado el problema con éxito, y las dificultades encontradas han servido para desarrollar habilidades que no se podrían haber obtenido en trabajos guiados.

## 5.3 Agradecimientos

Este trabajo no habría sido posible sin el apoyo y asesoramiento del tutor del TFG José Pelegrí Sebastià, profesor del departamento de Ingeniería Electrónica de la EPSG. Más que un tutor ha sido un mentor a lo largo del desarrollo de éste, tanto en el ámbito académico como en el personal.

También cabe mencionar y agradecer el asesoramiento de Jorge Alberola Lluch, Técnico Superior del GSyM del IGIC, el cual ha llevado la parte de hardware y medidas, y ha sabido orientar al alumno en las decisiones que ha tenido que tomar.

En lugar de un alumno con unos profesores orientándolo, este proyecto se ha realizado como un equipo, en el que siempre se podía contar con todos sus integrantes. Éste hecho ha ayudado sobremanera a la consecución de los objetivos, los cuales posiblemente no se hubieran podido alcanzar de la forma que se ha hecho en el caso de tener otro tipo de asesoramiento.

Finalmente, agradecer a familiares y amigos el apoyo y comprensión durante la dura tarea que ha supuesto la realización de este TFG. Muchísimas gracias a todos.

## 6. Referencias

- [1] «cran.r-project,» [En línea]. Available: <http://cran.r-project.org/web/packages/GeoLight/index.html>. [Último acceso: 26 05 2015].
- [2] N. Wählstrom, «Localization using Magnetometers and Light Sensors,» de *Localization using Magnetometers and Light Sensors*, Linköping, Suecia, Universidad de Linköping, departamento de Ingeniería Eléctrica, 2013, pp. 25-35.
- [3] O. M. Thomas Pfeleger, «Astronomy on the personal computer,» de *Astronomy on the personal computer*, Springer, 1999.
- [4] «NOAA,» [En línea]. Available: <http://www.esrl.noaa.gov/gmd/grad/solcalc/>. [Último acceso: 26 05 2015].
- [5] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Solar\\_cycle](http://en.wikipedia.org/wiki/Solar_cycle). [Último acceso: 16 05 2015].
- [6] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Metonic\\_cycle](http://en.wikipedia.org/wiki/Metonic_cycle). [Último acceso: 16 05 2015].
- [7] «Wikipedia,» [En línea]. Available: <http://en.wikipedia.org/wiki/Indiction>. [Último acceso: 16 05 2015].
- [8] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Julian\\_day](http://en.wikipedia.org/wiki/Julian_day). [Último acceso: 15 05 2015].
- [9] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Axial\\_tilt](http://en.wikipedia.org/wiki/Axial_tilt). [Último acceso: 16 05 2015].
- [10] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Mean\\_anomaly](http://en.wikipedia.org/wiki/Mean_anomaly). [Último acceso: 16 05 2015].
- [11] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Ecliptic\\_coordinate\\_system](http://en.wikipedia.org/wiki/Ecliptic_coordinate_system). [Último acceso: 16 05 2015].
- [12] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Right\\_ascension](http://en.wikipedia.org/wiki/Right_ascension). [Último acceso: 16 05 2015].
- [13] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Sidereal\\_time](http://en.wikipedia.org/wiki/Sidereal_time). [Último acceso: 2015 05 16].
- [14] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Hour\\_angle](http://en.wikipedia.org/wiki/Hour_angle).

- [15] «pladesemapesga.com/,» [En línea]. Available: <http://pladesemapesga.com/modules.php?op=modload&name=News&file=article&sid=28697>. [Último acceso: 24 05 2015].
- [16] M. W. Pedersen, Hidden Markov models for geolocation of fish, Technical University of Denmark, 2007.
- [17] e. a. Simeon Lisovski, «Geolocation by light accuracy and precision affected by environmental factors,» *Methods in Ecology and Evolution*, vol. 1, nº 3, pp. 603-612, 2012.
- [18] S. G. Hagen Stockhausen, star-oddi, Bergen, Noruega: Institute of Marine Research, 2009.
- [19] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Magnetic\\_dip](http://en.wikipedia.org/wiki/Magnetic_dip). [Último acceso: 20 05 2015].
- [20] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Magnetic\\_declination](http://en.wikipedia.org/wiki/Magnetic_declination). [Último acceso: 20 05 2015].
- [21] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Earth%27s\\_magnetic\\_field](http://en.wikipedia.org/wiki/Earth%27s_magnetic_field). [Último acceso: 20 05 2015].
- [22] «windows2universe.org,» [En línea]. Available: <http://www.windows2universe.org/earth/Water/temp.html&lang=sp>. [Último acceso: 25 05 2015].
- [23] «NOAA,» [En línea]. Available: <http://www.ospo.noaa.gov/Products/ocean/sst/contour/>. [Último acceso: 23 05 2015].
- [24] «Desert Star,» [En línea]. Available: <http://desertstar.com/product/seatag-geo/>. [Último acceso: 23 05 2015].
- [25] M. J. B. Roger D. Hill, Geolocation by Light Level, Redmond, USA: WildLife Computers, 2001.
- [26] «Wikipedia,» [En línea]. Available: [http://en.wikipedia.org/wiki/Equation\\_of\\_time](http://en.wikipedia.org/wiki/Equation_of_time). [Último acceso: 20 05 2015].