Desarrollo de una plataforma astroinformática para la administración y análisis inteligente de datos a gran escala

Registro de imágenes astronómicas

Victor Parada, Rodrigo Jara, Mauricio Solar, Marcelo Mendoza,
Luis Arévalo, Jorge Ibsen, Lars Nyman, Eduardo Vera, Diego Mardones,
Guillermo Cabrera, Paola Arellano, Karim Pichara, Paulina Troncoso,
Ricardo Contreras, Neil Nagar.

Santiago, 25 de noviembre de 2014

Resumen

El presente documento aborda el registro de imágenes aplicándolo a las ciencias astronómicas, donde se normalizarán galaxias que posean puntualmente forma elíptica.

Cada etapa de ajuste involucra un fundamento matemático, el cual será explicado mediante gráficas y posteriormente se indicarán los algoritmos involucrados que realizan el registro. Para finalizar se realizan una serie de pruebas que medirán la eficiencia, consistencia y el rendimiento del sistema generado.

Palabras Claves: registro de imágenes, normalización.

1. Resumen Ejecutivo

A partir del interés presentado por cinco universidades del país en el ámbito de la astronomía, nace la iniciativa de crear un observatorio virtual (CHIVO; Chilean virtual obsevatory) contando con el apoyo del observatorio ALMA, REUNA e instituciones privadas.

ALMA (Heisig, 2007), constantemente esta generando un gran volumen de datos, surge la necesidad entonces de desarrollar nuevas herramientas para analizar esta información y algoritmos que sean capaces de procesarlos.

Una de estas herramientas, que posee gran utilidad debido a sus diversos campos de aplicación, corresponde al registro de imágenes; técnica que se adapta dependiendo del área de aplicación y el objetivo de estudio, aunque dicha herramienta posee gran complejidad, ha presentado mucho interés pues existe una amplia bibliografía de diversos autores que la avalan.

El registro de imágenes consiste en ajustar geométricamente un conjunto de ellas, a partir de una imagen tomada como referencia, de modo que al procesarlas coincidan espacialmente en un plano.

En el caso de imágenes astronómicas, es muy limitada su obtención. Por un lado siempre los instrumentos de medición tendrán un margen de error que se tiene que considerar, como también, no siempre se contará con las mejores condiciones atmosféricas. Adicionalmente la cantidad de factores extraterrestres que pueden influir, como por ejemplo, una nebulosa obstaculizando el objeto de estudio, conlleva a mejorar las técnicas de observación, y el tratamiento de la información generada. El registro de imágenes, permite complementar la información existente para posteriormente analizarla.

Una aplicación importante dentro del registro de imágenes, es en el área de la medicina nuclear. Los avances tecnológicos que involucran las imágenes médicas, han permitido pasar de una radiografía convencional a una tomografía axial computarizada (CT), resonancia magnética nuclear (MRI), resonancia magnética funcional (fMRI), tomografía computarizada por emisión de fotón único (SPECT), o la tomografía por emisión de positrones (PET). El médico actual se da cuenta de las posibilidades que ofrece la combinación de múltiples imágenes, pero para llevarlo a cabo y poder compararlas directamente, es necesario eliminar las diferencias de tamaño, posicionamiento, orientación o incluso la distorsión espacial. Todo este proceso, es el que conoceremos como registro de imágenes. A grandes rasgos, el registro de imágenes se puede dividir en dos etapas: primero una transformación geométrica, con la intención de lograr la mayor concordancia entre ellas y segundo un proceso de fusión, que es la visualización conjunta de ellas.

Una aplicación del registro de imágenes en medicina, es el seguimiento de la evolución de una enfermedad en pacientes epilépticos. Gracias a esta técnica se pueden comparar estudios de un mismo paciente en diferentes momentos, pudiendo así cuantificar diferencias entre estudios ictales e inter-ictales con SPECT, y así localizar mejor el foco epiléptico (Lewis, et al., 2000). Este ajuste de imágenes, también conocido como proceso de normalización, permite comparar resultados entre pacientes, ajustando cada paciente a una plantilla común, de esta manera se puede realizar por ejemplo, un análisis estadístico sobre como afecta la enfermedad, como evoluciona en diferentes pacientes en el tiempo, o la efectividad del tratamiento.

Existen aplicaciones dentro del campo de la cartografía. Al sur de Orinoco, en Venezuela, existe un proyecto conocido como CARTOSUR (Miguel, 2003), el cual buscó determinar si la fusión de ortoimágenes

del radar SAR del proyecto CARTOSUR e imágenes Landsat Tematic Mapper, permiten lograr un mejor análisis visual de los elementos del territorio. La nubosidad propia de la región, impedía en muchos casos hacer vuelos fotogramétricos, o que estos lograrán un área de cobertura libre de nubes. Ambos tipos de imágenes tuvieron que pasar por filtros para poder combinarlas, por ejemplo, las imágenes de radar presentan un cierto tipo de 'ruido', conocido como moteado, el cual produce un efecto en la imagen de granularidad. En el caso de las imágenes de Landsat, presentan un desplazamiento en comparación a las orto-imágenes de radar (J., 1995), las cuales posteriormente tuvieron que ser corregidas. Finalmente se pudo concluir después de varias pruebas, que la fusión de estos dos tipos de imágenes ayudó definitivamente a identificar elementos de carácter antrópico y por lo tanto, apoyar las labores de captura de datos a partir de la interpretación de orto-imágenes de radar.

En este sentido el registro de imágenes cuenta con un conjunto de información que permite también el estudio a nivel astronómico, pudiendo abordar las características y variaciones de los elementos con los que se generarán estadísticas y estimaciones en el futuro.

La solución propuesta es desarrollar un módulo en Python (Rogan & Muñoz, 2012), utilizando matrices y transformaciones geométricas en el plano que normalicen imágenes en dos dimensiones. Para poder abrir, operar y escribir sobre cada archivo, se utilizará el módulo Astropy (Astropy) y para operaciones matemáticas generales el módulo math (Python). De esta manera realizar un ajuste de ellas en relación a su posición, tamaño, rotación y deproyección en un plano de dos dimensiones, generando una colección de imágenes a analizar.

Índice

Índice de figuras

Índice de algoritmos

2. Prototipo de solución

2.1. Definiciones básicas

Para llevar a cabo la normalización se han definido cuatro etapas (Fig. 1) en las que son ajustadas un conjunto de imágenes a un formato general antes de ser analizadas. Se entiende que en cada imagen existe el mismo objeto de estudio, pero puede variar su posición, tamaño y forma. Al ir procesando cada imagen, se van aplicando los filtros respectivos, los cuales permiten que las imágenes queden con un patrón común que facilite su estudio. El siguiente esquema visualiza cómo se desarrolla el proceso de normalización en un conjunto de imágenes:

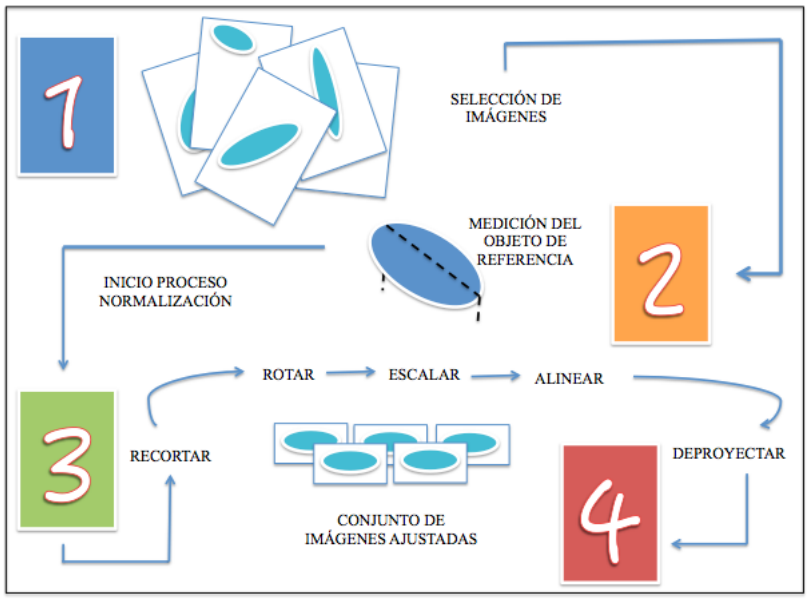

Figura 1: Representación del proceso para registrar una imagen.

I El diagrama parte con la selección de imágenes que poseen objetos que al usuario le interesa estudiar. Una de estas imágenes será tomada como referencia para realizar el ajuste de las restantes.

II Cada vez que una imagen es leída por el programa, éste realiza la toma de medidas, es decir, la longitud del eje mayor del objeto y su ángulo de proyección respecto a un eje horizontal.

III Luego, pasa por cinco etapas de ajuste, extracción del objeto de interés, rotación del objeto con respecto al eje mayor, el cual se deja sin pendiente, luego se escala y se alinea en relación a la imagen de referencia,

IV para que finalmente, se pueda deproyectar, transformando la elipse en un círculo.

Cada nivel de ajuste contempla el ingreso de información externa respecto a la posición actualizada del objeto, es decir, se necesita saber antes y después de realizar el ajuste en qué coordenadas va quedando

el objeto. Cabe destacar que todos los algoritmos realizados para cada etapa de ajuste, exceptuando la deproyección, pueden ser empleados en cualquier tipo de objeto astronómico, ya sean galaxias, cúmulos globulares, nebulosas, etcétera, y para fines de esta memoria se trabajó puntualmente en la deproyección de objetos elípticos.

2.2. Preparación de la información

Corresponde en esta instancia seleccionar diversos objetos con forma similar a una elipse, a los cuales se les realizan transformaciones geométricas en el plano (Fundación Polar). En este proceso, se selecciona una imagen de referencia con la cual se trabaja para llevar a cabo la normalización. Los objetos elípticos a manipular corresponden a galaxias que presentan una forma elíptica.

Es esencial conocer en cada etapa las coordenadas que componen el borde de la elipse, lo que ayuda a procesar la información útil dentro de la imagen, determinar la medida del eje mayor e inclinación de esta pseudo elipse. Es aquí donde se puede determinar que tan grande o tanto más pequeña es del resto, como también saber su ángulo de rotación respecto al eje horizontal y su medida de deproyección, es decir, lo que le falta a la elipse para transformarla en círculo.

Entiéndase que una elipse es una curva simétrica cerrada, que se obtiene al cortar la superficie de un cono con un plano oblicuo al eje de simetría, en palabras mas sencillas, es como un circunferencia estirada simétricamente. En una elipse, se puede identificar el eje mayor, como los puntos que están más distantes entre sí, los cuales se determinan calculando las diferencias que existen entre los puntos correspondientes al borde de la elipse. Para saber la distancia entre dos puntos en un plano, es utilizado el "Teorema de Pitágoras" (1), el cual dice, que la suma de los catetos al cuadrado es igual a la hipotenusa al cuadrado, que en este caso es la distancia entre dos puntos de la elipse, como lo muestra la Fig. 2.

$$a^2 = b^2 + c^2 \tag{1}$$

Matemáticamente, la expresión que calcula la distancia entre dos puntos se escribe (2):

$$\text{eje mayor} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

Conociendo la distancia del eje mayor, se procede a determinar su ángulo en relación al eje horizontal del plano. Esto, permite reconocer en que medida rotar el objeto antes de ser comparado. Para encontrar el ángulo de rotación, basta calcular la inversa de la función tangente (3), en la que, aplicando el arco tangente a la razón existente entre la diferencia de los puntos en el plano vertical, dividida por la diferencia de los puntos en el plano horizontal, da como resultado, el ángulo de la recta. Gráficamente se puede apreciar en la Fig. 3.

$$\partial = \tan^{-1}\left(\frac{Y_{2-Y_1}}{X_{2-X_1}}\right) \tag{3}$$

El punto centro de la elipse, permite ajustar el resto de los objetos a ese punto de referencia. Para definir el punto, basta con dividir el eje mayor en dos y estimar a que coordenada corresponde. Para identificar la posición que ocupa en el plano el centro del objeto, se le suma a una de las coordenadas, que pertenece a la esquina del eje mayor, la proyección de la mitad de este eje.

Para saber la posición en el plano del centro del objeto se toma una esquina del eje mayor como referencia y se le suma tanto en el eje x (eje horizontal) como en el eje y (eje vertical) la proyección de la mitad del eje mayor. La expresión matemática (4) y (5) corresponden al cálculo de la distancia en los dos ejes del plano.

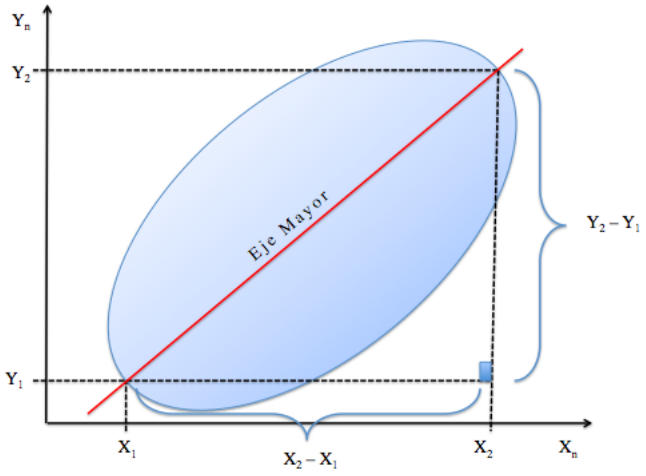

Figura 2: Cálculo del eje mayor de la elipse.

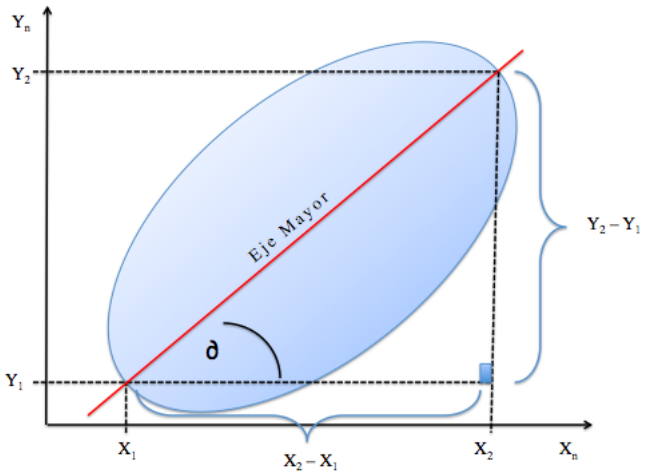

Figura 3: Determinación del ángulo de proyección del eje mayor de la elipse con respecto al eje horizontal del plano.

$$\text{distancia en eje } x = \cos(\partial) * \text{ distancia eje mayor } * 0{,}5 \tag{4}$$

$$\text{distancia en eje } y = \sin(\partial) * \text{ distancia eje mayor } * 0{,}5 \tag{5}$$

Una vez reconocida la distancia del eje respecto al punto menor ya es posible determinar la coordenada del punto central. La expresión (6), suma estas distancias a la coordenada menor del eje obteniendo así el punto central del objeto.

$$\text{punto central eje} = (X_1 + \text{ distancia en eje } x, Y_1 + \text{ distancia en eje } y) \tag{6}$$

Gráficamente, lo podemos apreciar en la Fig. 4.

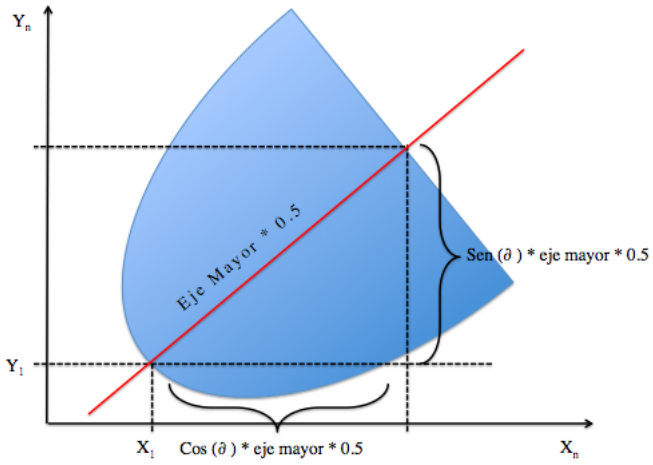

Figura 4: Cálculo del punto central de la elipse.

Los mecanismos antes expuestos desde el punto de vista algorítmico se detallan en la siguiente sección.

2.3. Prototipado para proceso de normalización

La normalización define cinco etapas:

- Extraer de la imagen el objeto de interés.
- Rotar el objeto dejándolo paralelo al eje horizontal.
- Escalar el objeto dejándolo del mismo tamaño que el objeto de referencia.
- Alinear el objeto a un punto de referencia común.
- Deproyectar el objeto hasta que quede como un círculo.

Algoritmo recorte Para extraer de la imagen el objeto de interés, descartando todo el resto de la información que no es necesaria, se debe recortar la imagen y así trabajar solo con la información útil, lo que permite realizar sólo los cálculos necesarios, entregando la información más precisa. Al realizar la observación de una determinada ubicación en el espacio, puede ocurrir que la imagen posea más de un objeto, pero el foco de interés radica en uno solo de ellos, en este caso, es sumamente útil individualizar y concentrar el estudio en la información que se necesita descartando el resto. Para recortar la imagen se debe determinar los puntos extremos del cuerpo elíptico, es decir, mirándolo desde el punto matricial, encontrar el mayor y el menor valor de la fila y columna. Para esto, se recorre el borde del objeto hasta encontrar los cuatro puntos. El algoritmo (1) que se presenta a continuación realiza la extracción.

En las líneas 2, 3, 4 y 5, se declaran las variables que guardarán el menor y mayor valor entre las filas y columnas que pertenecen al borde del objeto de estudio, nótese que no es identificar el valor existente en la posición, sino la posición en si. De la línea 11 a la 21, se encuentra el valor para fila_mayor, fila_menor, columna_mayor y columna_menor. Finalmente de la línea 24 a la 36, se traslada el objeto de manera tal, que los valores encontrados para fila y columna, sean el borde la imagen.

Algoritmo de rotación 2 La rotación consiste en girar un objeto en torno a punto. Por cada imagen que ingresa se calcula su ángulo de proyección respecto al eje horizontal para que posteriormente se rote en esta medida hasta que el ángulo sea cero, dado que es más sencillo dejar bajo la misma pendiente que tratar de hacer coincidir a una de referencia.

Como en la toma de medidas, el ángulo está determinado, en esta etapa se da la orden de rotar en sentido antihorario el complemento del ángulo.

Una imagen es básicamente una matriz de n filas y m columnas, con posiciones del $(0,0)$ al $(n-1, m-1)$. Para poder rotarla se multiplica una matriz de rotación definida con seno y coseno por cada coordenada. Antes de rotar la imagen, existen algunos ángulos (90, 180 y 270°), en donde no es necesario multiplicar las coordenadas de la matriz imagen por la matriz de rotación, tan solo es necesario hacer una traslación de puntos.

Cuando el ángulo a rotar es diferentes a los antes expuestos, recién se utiliza la matriz de rotación. Cabe destacar, que la rotación se realiza en torno al origen, es decir, el punto $(0,0)$, entonces antes de girar la imagen, se debe trasladar el centro del objeto a ese punto. Esto ocurre básicamente por la forma en que está definida la matriz de rotación (Ec. 7).

$$\begin{pmatrix} \cos\partial & -\sin\partial \\ \sin\partial & \cos\partial \end{pmatrix} \tag{7}$$

Para evitar que alguna posición de la matriz quede negativa, se traslada el centro del objeto al origen, se rota y se vuelve a dejar en un cuadrante positivo. Para esto se debe saber cual es el centro del objeto, de modo de desplazar todos los puntos en esa medida, luego se rota identificando el punto más negativo. Esta medida sirve para desplazar en esa cantidad todas las coordenadas a un eje positivo.

El Alg. 3 utiliza variables como; coseno y seno, que corresponden al cálculo trigonométrico del ángulo ingresado; punto_central_x y punto_central_y para indicar cual es el centro del objeto; distancia_centro

Algorithm 1 Algoritmo recorte.

```
Función recortar(matriz, borde):
fila_mayor = 0
file_menor = borde[0]
columna_mayor = 0
columna_menor = borde[1]
i = 0
while i! = largo(borde) - 1 do
    if i es impar then
        continúe
    end if
    if fila_mayor < borde[i] then
        fila_mayor = borde[i]
    end if
    if fila_menor > borde[i] then
        fila_menor = borde[i]
    end if
    if columna_mayor < borde[i + 1] then
        columna_mayor = borde[i + 1]
    end if
    if columna_menor > borde[i + 1] then
        columna_menor = borde[i + 1]
    end if
end while
i = 0
while i! = largo(borde) - 1 do
    if i es impar then
        continúe
    end if
    borde[i] = borde[i] - fila_menor
    borde[i + 1] = borde[i + 1] - columna_menor
    i = fila_menor
end while
while i! = fila_mayor + 1 do
    j = columna_menor
    while j! = columna_mayor + 1 do
        matriz_final[i - fila_menor][columna_menor] = matriz[i][j]
    end while
end while
retorno matriz_final
```

Algorithm 2 Algoritmo de rotación.

```
Funcion rotar (matriz, NAXIS1, NAXIS2, ángulo):
if ángulo > 360 OR ángulo < 1 then
    imprimir "<Error: Imagen no rotada, ángulo no permitido>"
    retorne matriz
end if
#—— PARA 0 NO ES NECESARIO ROTAR ——#
if ángulo == 0 OR ángulo == 360 then
    retorne matriz
end if
#—— PARA 90, 180 Y 270 ES UNA SIMPLE TRASLACIÓN DE PUNTOS ——#
if ángulo == 90 then
    matriz_final = np.zeros(NAXIS2, NAXIS1)
    while i in range(NAXIS1) do
        while j in range(NAXIS2) do
            matriz_final[NAXIS2 - j - 1][i] = matriz[i][j]
        end while
    end while
    retorne matriz_final
end if
if ángulo == 180 then
    matriz_final = np.zeros(NAXIS1, NAXIS2)
    while i in range(NAXIS1) do
        while j in range(NAXIS2) do
            matriz_final[NAXIS1 - i - 1][NAXIS2 - j - 1] = matriz[i][j]
        end while
    end while
    retorne matriz_final
end if
if ángulo == 270 then
    matriz_final = np.zeros(NAXIS2, NAXIS1)
    while i in range(NAXIS1) do
        while j in range(NAXIS2) do
            matriz_final[j][i] = matriz[i][j]
        end while
    end while
    retorne matriz_final
end if
```

que corresponde a la distancia del origen al centro del objeto; porcentaje_columna_derecha, porcentaje_columna_izquierda, porcentaje_fila_abajo y porcentaje_fila_arriba, que guardan proporcionalmente el valor asignado a una posición decimal. Su fin es determinar la medida del arreglo bidimensional que contendrá la imagen de la matriz rotada.

Cuando una imagen es rotada, existen puntos que quedan situados en pixeles intermedios. Para no dajar puntos vacíos, cuando un punto queda en una posición decimal, se guarda el valor en la parte entera de la coordenada y en la parte entera redondeada, pero en cada una un porcentaje. Lo que corresponde a dejar, del valor, en cada posición, depende del valor decimal. Por ejemplo, si deseo guardar el valor 1032 en la posición (0, 2.8), se guardará en la posición (0, 3) el 80 % de 1032, y en (0, 2) el 20 % . Si se ubica el número 2.8 en una recta numérica, se identifica que el valor esta cercano a 3.0, y distante de 2.0, gráficamente se ve en la Fig. 5.

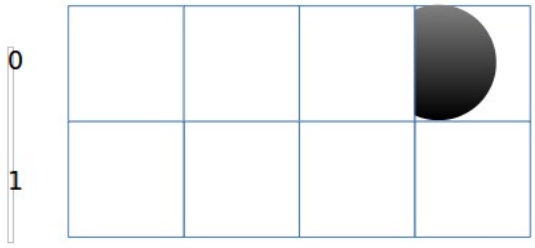

Figura 5: Imagen rotada con puntos en pixeles intermedios.

Si el punto cae entre cuatro posiciones de la matriz, el mecanismo para calcular el porcentaje es similar. Por ejemplo, si en la coordenada (2.7, 1.1) se quiere guardar el valor 2021, este valor queda repartido entre las coordenadas (2, 1), (2, 2), (3, 1) y (3, 2). El porcentaje asignado se calcula de la siguiente manera:

a Ver el porcentaje a asignar por filas y por columnas, es decir, si analizamos la coordenada (2.7, 1.1) desde el punto de vista fila, tendremos un 70 % de 2021 en la fila 3, y un 30 % para la fila 2, y si vemos el porcentaje respecto a las columnas, tendremos un 10 % de 2021 para la columna 2, y un 90 % para la columna 1.

b Finalmente cruzar la información, y la multiplicación de esos porcentajes, coorresponde a la asignación de 2021 en esa posición.

Calculando lo antes expuesto se tiene que:

- El 30 % del 90 % de 2021 se guarda en la posición (2, 1), que es igual a 545.67.
- El 30 % del 10 % de 2021 se guarda en la posición (2, 2), que es igual a 60.63.
- El 70 % del 90 % de 2021 se guarda en la posición (3, 1), que es igual a 1273.23.
- El 70 % del 10 % de 2021 se guarda en la posición (3, 2), que es igual a 141.47.

Algorithm 3 Algoritmo para determinar la medida del arreglo bidimensional que contendrá la imagen de la matriz rotada.

```
coseno = math.cos((angulo*math.pi)/180)
seno = math.sin((angulo*math.pi)/180)
punto_central_x = int(round(NAXIS1/2))
punto_central_y = int(round(NAXIS2/2))
#—— Traslación del centro objeto al origen ——#
distancia_centro = distancia(0, 0, punto_central_x, punto_central_y)
#—— Se busca el punto mas negative y positive entre filas y columnas ——#
vec = [0, 0, NAXIS1, NAXIS2, NAXIS1, 0, 0, NAXIS2]
fila_mas_negativa = columna_mas_negativa = 0
fila_mas_positiva = columna_mas_positiva = 0
while i in range(7) do
    alfa = (vec[i] - distancia_centro) * coseno - (vec[i + 1] - distancia_centro) * seno
    beta = (vec[i] - distancia_centro) * seno + (vec[i + 1] - distancia_centro) * coseno
    if alfa < fila_mas_negativa then
        fila_mas_negativa = int(math.ceil(alfa))
    end if
    if alfa > fila_mas_positiva then
        fila_mas_positiva = int(math.ceil(alfa))
    end if
    if beta < columna_mas_negativa then
        columna_mas_negativa = int(math.ceil(beta))
    end if
    if beta > columna_mas_positiva then
        columna_mas_positiva = int(math.ceil(beta))
    end if
    distancia_1 = fila_mas_positiva + abs(fila_mas_negativa)
    distancia_2 = columna_mas_positiva + abs(columna_mas_negativa)
    while x in range(NAXIS1) do
        while y in range(NAXIS2) do
            #—— a x e y hay que restarle y luego sumarle la traslacion ——#
            a = ((x - distancia_centro) * coseno
            a = a - (y - distancia_centro) * seno) + fila_mas_negativa
            b = ((x - distancia_centro) * seno
            b = b + (y - distancia_centro) * coseno ) + abs(columna_mas_negativa)
        end while
    end while
end while
```

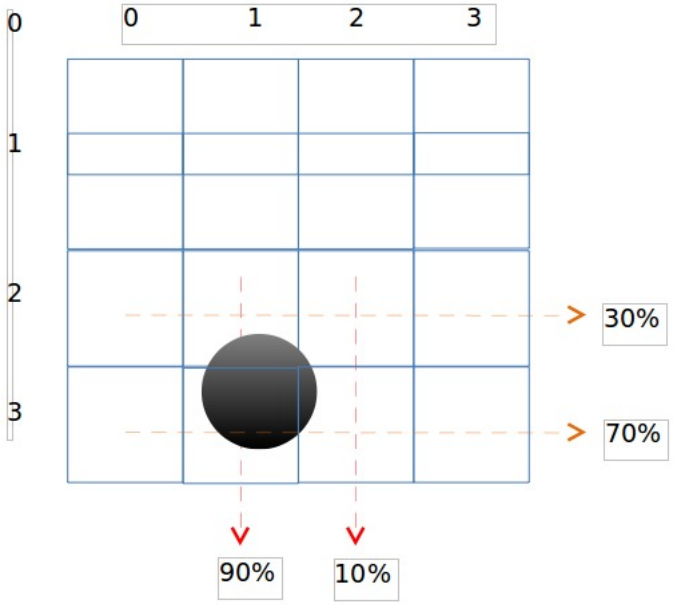

Figura 6: Figura rotada con puntos entre 4 posiciones.

Sumando las cantidades asignadas podemos notar que da 2021. La Fig. 6, refleja gráficamente esta situación.

El sentido de repartir la información, es que en un arreglo bidimensional no existen posiciones con valores decimales, entonces se debe distribuir proporcionalmente la información entre posiciones enteras. Puede ocurrir, que en la posición que se asigna el valor, tenga previamente información guardada, en este caso tan solo se adiciona.

El Alg. 4 en pseudo código determina los porcentajes asignados según las filas y columnas que contienen al punto.

Hasta el momento, solo se ha extraído información de la coordenada rotada que se quiere posicionar, es decir, los porcentajes involucrados en la repartición. El código restante (5) realiza el cruce de información según la cantidad de valores decimales contenidos en la posición, por ejemplo, si el valor 3247 se quiere guardar en la coordenada (a, b), el algoritmo determinará en cuantas posiciones repartir el valor según a o b sean decimales.

La base de este algoritmo es identificar que posiciones, al rotar la imagen, quedan en un valor decimal, de esta manera se ve en cuantas casillas repartir el valor a posicionar. Para esto se definen dos banderas, cada una con un valor 100, entonces cuando el valor en x es decimal se le asigna el valor 101 a la 'bandera_x', si el valor de y es decimal se le asigna a la 'bandera_y' el valor 110, de esta manera si la suma de las bandera da 201, significa que tan solo x es decimal, si suman 210 significa que tan solo y es decimal, si suman 211 significa ambos son decimales y si suman 200 significa que ambos son números enteros. De esta manera se sabe cuantas casillas se le asigna un porcentaje del valor a guardar.

Algorithm 4 Algoritmo para determinar la medida del arreglo bidimensional que contendrá la imagen de la matriz rotada.

```
for x in range(NAXIS1) do
    for y in range(NAXIS2) do
        bandera_decimal_a = 100
        bandera_decimal_b = 100
        #—— Los valores de a y b, corresponden a la coordenada (x,y) rotada ——#
        if a - int(a) ! = 0 then
            bandera_decimal_a = 101
        end if
        if b - int(b) ! = 0) then
            bandera_decimal_b = 110
        end if
        #—— Esto es un swith() hecho artesanalmente ——#
        suma_banderas = bandera_decimal_a + bandera_decimal_b
        #—— abs() es una function que calcula el valor absolute ——#
        #—— int() calcula el valor entero de un número ——#
        while (1) do
            porcentaje_columna_derecha = porcentaje_columna_izquierda = 0
            porcentaje_fila_abajo = porcentaje_fila_arriba = 0
            porcentaje_fila_arriba = abs(abs(a) - int(abs(a)))
            porcentaje_fila_abajo = 1 - porcentaje_fila_arriba
            porcentaje_columna_derecha = abs(abs(b) - int(abs(b)))
            porcentaje_columna_izquierda = 1 - porcentaje_columna_derecha
        end while
    end for
end for
```

Algorithm 5 Algoritmo para realizar el cruce de información según la cantidad de valores decimales contenidos en la posición.

```
#—— Solo A es decimal. Ceil (), función que redondea al entero siguiente ——#
if suma_banderas == 201 then
    matriz_final[int(a)][b] += porcentaje_fila_abajo * matriz[x][y]
    matriz_final[ceil(a)][b] += porcentaje_fila_arriba * matriz[x][y]
    break
end if
#—— Solo B es decimal ——#
if suma_banderas == 210 then
    matriz_final[a][int(b)] += porcentaje_columna_izquierda * matriz[x][y]
    matriz_final[a][ceil(b)] += porcentaje_columna_derecha * matriz[x][y]
    break
end if
#—— Ambos son decimales ——#
if suma_banderas == 211 then
    matriz_final[int(a)][int(b)] += porcentaje_fila_abajo * porcentaje_columna_izquierda * matriz[x][y]
    matriz_final[ceil(a)][ceil(b)] += porcentaje_fila_arriba * porcentaje_columna_derecha * matriz[x][y]
    matriz_final[int(a)][ceil(b)] += porcentaje_fila_abajo * porcentaje_columna_derecha * matriz[x][y]
    matriz_final[ceil(a)][int(b)] += porcentaje_fila_arriba * porcentaje_columna_izquierda * matriz[x][y]
    break
end if
#—— Ambos son enteros #——
if suma_banderas == 200 then
    matriz_final[a][b] = matriz[x][y]
    break
end if
retorne matriz_final
```

Algoritmo de escalamiento (Alg. 8) Escalar es agrandar o achicar una imagen en un factor determinado. Para escalar una imagen se necesita saber la razón de escala, es decir, cuánto más grande o más pequeña es el eje mayor de la imagen actual con respecto a la imagen de referencia. Una vez identificado el valor, se procede a multiplicar cada coordenada de la matriz por este factor (Ec. 8), permitiendo distribuir los puntos uniformemente dentro del plano. Si el factor de escala es un número entero, quedan puntos intermedios sin información, dado esto, deben ser formados a partir de los cuatro puntos mas cercanos. En caso de que el propósito sea achicar la imagen, no será necesario aplicar lo antes expuesto.

$$[X_2, Y_2] = [X_1 * \text{factor}, Y_2 * \text{factor}] \tag{8}$$

El algoritmo recorre cada posición de la matriz y en la posición resultante, que es la posición original por el factor de escala, guarda el valor de la posición original. Esto se realiza siempre y cuando se necesite expandir la imagen, si el tamaño actual es igual al original, no es necesario llevar a cabo esta tarea.

Algorithm 6 Algoritmo de escalamiento.

```
1: Función escalar(matriz, NAXIS1, NAXIS2, razón):
2: if razón == 1 then
3:    retornar matriz
4:    while x distinto NAXIS1 do
5:       while y distinto NAXIS2 do
6:          matriz_final[x * razón][y * razón] = matriz[x][y]
7:       end while
8:    end while
9: end if
```

Luego de escalar la imagen y solo en caso de que se haya agrandado, se debe interpolar, es decir, completar la información faltante a partir de los puntos cercanos. Los puntos a considerar no se ven respecto al punto vacío, sino que a partir de los cuatro puntos con información se rellenan todos los puntos intermedios (Ec. 9), esto es mas sencillo de diseñar, ya que desde un punto con información a otro, la distancia es igual al valor de la razón de escala. El valor asignado a la posición sin información corresponde a un porcentaje de los cuatro vecinos con información mas cercanos, donde la distancia determina su peso. Por ejemplo, si se tuviera cuatro cargas eléctricas (a, b, c y d) formando un cuadrado (todas de igual magnitud), y un ente conductor se somete a este campo magnético, dependiendo de donde este ubicado el ente, se puede determinar la fuerza con que es atraido por las cargas. Por ejemplo, en la Fig. 7, hay un cilindro centrado en una superficie plana, entre las cargas Q1, Q2, Q3 y Q4. En este caso, cada carga influye un 25 % de su fuerza sobre el ente.

$$\text{valor} = (x, y) * a + (x + \text{razón}, y) * b + (x, y + \text{razón}) * c + (x + \text{razón}, y + \text{razón}) * d \tag{9}$$

Si el ente se desplaza a otra posición, la forma en que cada carga influye sobre el ente cambia y se debe considerar la distancia como una referencia. La Fig.8, muestra un ente que está muy apegado a la

carga Q4. En este caso, la carga Q4 influye mayoritariamente sobre el ente, en menor grado las cargas Q2 y Q3, y finalmente, con un porcentaje muy bajo la carga Q1.

Para determinar el porcentaje asignado a cada carga, se suman todas las distancia entre cada carga y el ente, y luego se divide cada distancia individual por el total. De esta manera, el valor encontrado, se le asigna a la carga del frente. Por ejemplo, si la suma de todas las distancia individuales da 10 unidades, y la distancia individual de Q1 al ente es 6 unidades, significa que Q4 influye un 60 % sobre el ente.

Para el ejemplo expuesto, cada carga corresponde a un vecino con información, y el ente es una posición de la matriz sin información. Como la imagen posee puntos con información igualmente distribuida, los porcentajes determinados para las posiciones sin información para los cuatro primeros vecinos, es constante para el resto de la imagen. En la Fig.9, se tiene una matriz con un punto ubicado en la posición (0, 1), el cual posee un x1 % de h1, un x2 % de h2, un x3 % de h3 y un x4 % de h4. Para todos los puntos ubicados en la posición $(0, 2i)$, con i pertenecientes a los enteros positivos, le corresponderá la misma proporción.

El Alg. 8 muestra la manera en que se realiza la interpolación.

El Alg. 8 posee dos partes vitales, primero, de la linea 7 a la 11, determina como queda particionada la posición sin información, y segundo, de la linea 15 a la 31, extiende y aplica este cáculo al resto de la información.

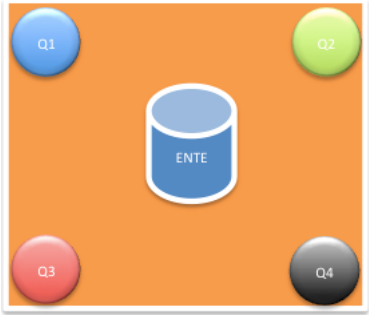

Figura 7: Ente conductor sometido a 4 cargas eléctricas de igual magnitud.

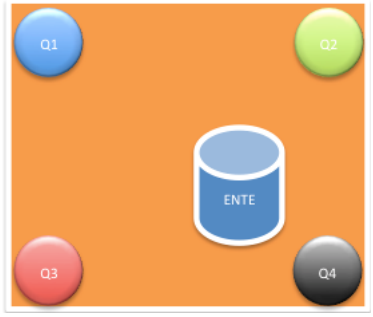

Figura 8: Ente conductor sometido a 4 cargas eléctricas de igual magnitud.

Algorithm 7 Algoritmo de interpolación.

```
1: porcentajes = [ ] #—— Vector dinámico ——#
2: while i in range(razon + 1) do
3:   while j in range(razon + 1) do
4:     if i == 0 AND j == 0 OR i == 0 AND j == razón OR i == razón AND j == 0 OR i == razón AND j == razón then
5:       continue
6:     else
7:       suma_distancias = info_imagen.distancia(i,j,0,0) + info_imagen.distancia(i,j,razon,0) + info_imagen.distancia(i,j,0,razon) + info_imagen.distancia(i,j,razon,razon)
8:       porcentajes.append(info_imagen.distancia(i,j,razon,razon)/suma_distancias)
9:       porcentajes.append(info_imagen.distancia(i,j,razon,0)/suma_distancias)
10:      porcentajes.append(info_imagen.distancia(i,j,0,razon)/suma_distancias)
11:      porcentajes.append(info_imagen.distancia(i,j,0,0)/suma_distancias)
12:    end if
13:  end while
14: end while
15: posicion_vector = 0
16: while x < (NAXIS1 - 1) * razón do
17:   while y < (NAXIS2 - 1) * razón do
18:     pos = [x, y, x + razon, y, x, y + razon, x + razon, y + razon]
19:     while i in range(x, x + razon + 1) do
20:       while j in range(y, y + razon + 1) do
21:         if i == pos[0] AND j == pos[1] OR i == pos[2] AND j == pos[3] OR i == pos[4] AND j == pos[5] OR i == pos[6] AND j == pos[7] then
22:           continue
23:         end if
24:         matriz_final[i][j] = matriz[x][y] * porcentajes[posicion_vector] + matriz[x][y + razon - 1] * porcentajes[posicion_vector+1] + matriz[x + razon - 1][y] * porcentajes[posicion_vector+2] + matriz[x + razon - 1][y + razon - 1] * porcentajes[posicion_vector+3]
25:         posicion_vector += 4
26:       end while
27:     end while
28:     y = y + razón
29:   end while
30:   y = 0
31:   x = x + razón
32: end while
33: retorne matriz_final
```

Algoritmo de alineación Alinear geométricamente representa el desplazamiento de un punto o un conjunto de puntos según un vector fijo no nulo. El proceso de alinear consiste en llevar el centro del objeto de estudio a punto de referencia común, con el fin que todos los objetos en cada imagen tenga la misma posición antes de ser analizados. Como la imagen que está siendo procesada ya posee el mismo tamaño y rotación que la de referencia, se puede trasladar tranquilamente el objeto a ese punto, sin preocuparse de que la información quede fuera de los márgenes.

Si la imagen de referencia es más pequeña que la actual, al desplazar los puntos, solo aquellos que estén dentro de la dimensión de la imagen de referencia quedarán guardados. El resto, al no poseer información, serán descartados automáticamente. Cuando la imagen de referencia es más grande, se traspasará la información completamente sin descartar puntos.

Para determinar el vector de traslación, se calcula la distancia en x y en y, entre el punto de referencia y el punto centro actual, de esta manera, los valores son desplazado en esa distancia. La Fig. 10 grafica un ejemplo de un punto que debe ser desplazado 1 unidades en x, y 2 unidades en y.

La función alinear no necesita interpolar la información dado que el vector de traslación, tiene valores enteros, entonces la posición resultante siempre cae en una posición entera.

Algoritmo de deproyección Es el proceso por el cual se cambia la perspectiva en que se ve un objeto. Si pusiéramos las imágenes en un plano tridimensional (x, y, z), siendo z la coordenada que da la profundidad, se apreciaría que el ángulo de inclinación respecto al eje varía en cada imagen, lo que también se podría llamar perspectiva. Para poder comparar correctamente las imágenes es necesario dejar todos los objetos paralelos al plano (x, y).

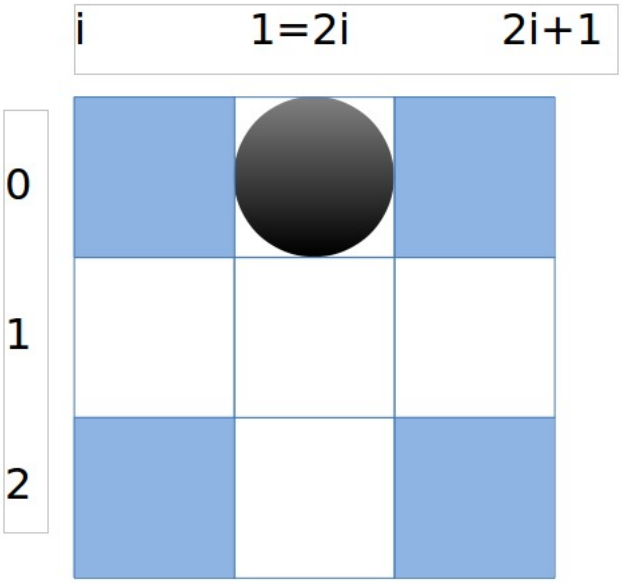

Figura 9: Matriz ubicada con un punto en la posición (0, 1).

Algorithm 8 Algoritmo de interpolación.

```
Función alinear(x1, y1, x2, y2, dim_1_x, dim_1_y, dim_2_x, dim_2_y, actual):
diferencia_x = x1 - x2
diferencia_y = y1 - y2
while i distinto dim_2_x do
   while j distinto dim_2_y do
      x = i + diferencia_x
      y = j + diferencia_y
      if x > 0 AND x < dim_1_x AND y > 0 AND y < dim_1_y) then
         matriz_final[x][y] = actual[i][j]
      end if
   end while
end while
retornar matriz_final
```

La Fig. 11 y la Fig. 12 muestra como dos galaxias espirales recrean esta misma situación, por un lado una está más recostada, destacando más su canto, en cambio en la otra, es mas visible su centro y su extensión.

Para corregir esta situación, es necesario dejar ambos objetos bajo el mismo enfoque, es decir, cambiar el ángulo del objeto en relación a un tercer eje. Lamentablemente esto no es factible, dado que no existe una tercera coordenada. Para lograr un efecto similar, la elipse se puede 'estirar' de forma tal, que manteniendo el mismo eje mayor de la elipse, se trasforme en un círculo, es decir, encontrar una relación entre un punto de la elipse y un punto del círculo. En la Fig. 13 tenemos un círculo con una elipse inscrita, compartiendo el mismo diámetro.

Podemos notar que tanto $(X1, Y1)$ e $(X2, Y2)$ comparten el mismo ángulo, es decir sabiendo el ángulo compartido y la distancia de cada punto al centro geométrico del cuerpo, se podría determinar la coordenada equivalente en el circulo. Como el punto $(X1, Y1)$ y el centro (h, k) es conocido, aplicando propiedades trigonométricas sencillas se puede obtener el ángulo.

Supongamos que la distancia del punto (h, k) a un punto (X, Y) de la elipse mide a, y trazando una

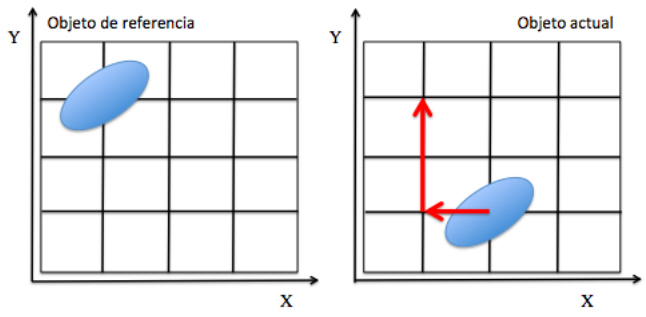

Figura 10: Traslación de un objeto en el plano (x, y)

perpendicular desde ese punto al eje horizontal de largo $Y–k$, se estaría formando un triángulo rectángulo donde $Y–k$, es el cateto opuesto y a, es la hipotenusa, situación graficada en la Fig. 14.

Para determinar el ángulo ∂, se calcula el arcoseno de la razón entre el cateto opuesto al ángulo con la hipotenusa (Ec. 10).

$$\partial = \sin^{-1}\left(\frac{Y-k}{a}\right) \tag{10}$$

Este ángulo es vital para encontrar el punto equivalente en el círculo, ya que teniendo la distancia, que

Figura 11: Galaxia espiral Andrómeda

Figura 12: Galaxia espiral “sombrero”

en el caso de un punto borde es igual al radio, y escribiendo el punto del círculo en forma paramétrica se puede obtener la coordenada. La expresión de a continuación (Ec. 11), representa la forma paramétrica de un punto (x, y).

$$(x, y) = (dcos(x), dsen(x)) \tag{11}$$

Remplazando el ángulo encontrado a partir del punto en la elipse y la distancia del centro del cuerpo al punto proyectado, se obtiene la coordenada dentro del círculo. Esta distancia varía según el punto, pero se determina en relación al punto mayor que está a distancia de un radio. Lamentablemente, este escenario cambia un poco en el ámbito computacional, ya que el primer cuadrante del plano cartesiano

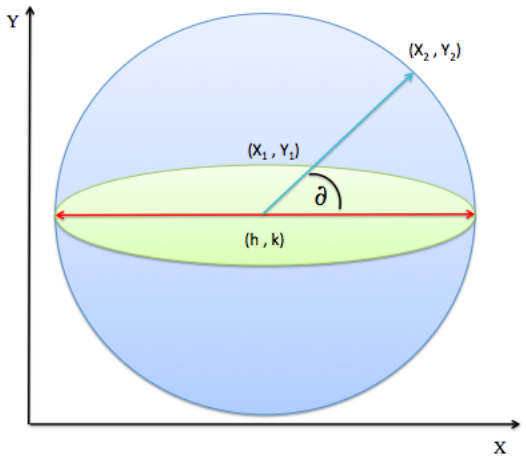

Figura 13: Proyección de un punto elipse a la circunferencia.

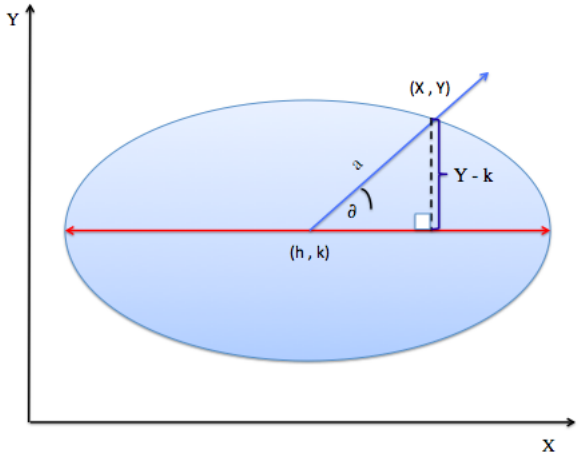

Figura 14: Catetos e hipotenusa de un triangulo rectángulo inscrito en una elipse.

no es equivalente al plano matricial de una imagen, sino más bien es similar. Primero, el plano cartesiano tiene el punto (0,0) en el origen, en cambio en una imagen el (0,0) se encuentra en la primera posición de la esquina superior de la matriz, como lo vemos en la Fig. 15 y Fig. 16.

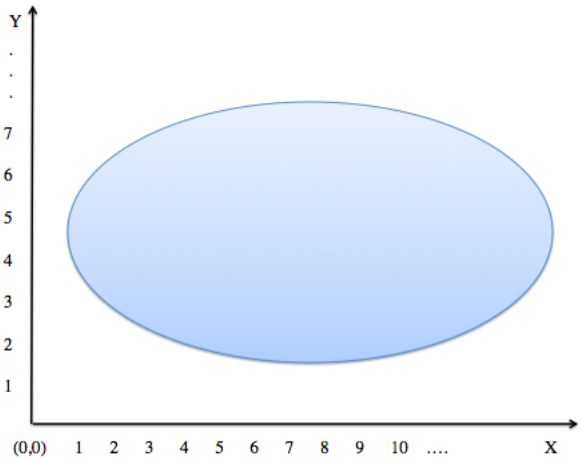

Figura 15: Elipse posesionada en el primer cuadrante de un plano cartesiano.

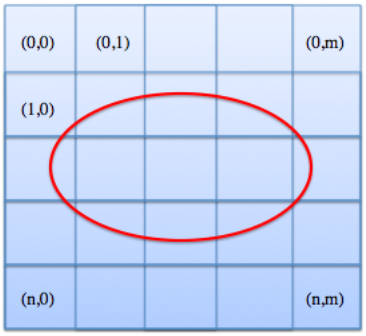

Figura 16: Representación matricial de una elipse en un arreglo bidimensional.

Esto cambia completamente la forma de aplicar las funciones trigonométricas al plano normal dado que la coordenada (x, y) representada en una "matriz computacional" se lee (y, x), entonces la expresión paramétrica del círculo también se escribe al revés (Ec. 12). Otro aspecto a considerar, es la forma de leer el ángulo, en el plano cartesiano se hace en sentido antihorario, Fig. 17, pero en este caso se hace en sentido horario, 18, también por el mismo motivo.

$$(x, y) = (dsen(x), dcos(x)) \tag{12}$$

Teniendo claro esta relación ya es posible proyectar los puntos correctamente. El Alg. 9 que se presenta, refleja la manera en que se elije un punto de la elipse y en base al ángulo que posee se proyecta hacia

el círculo. Éste recibe como parámetro de entrada el centro geométrico del objeto, la dimensión de la imagen NAXIS1, NAXIS2, un vector con información de las coordenadas del borde del objeto, la matriz de información del objeto y el radio de la circunferencia, que sería la mitad del eje mayor de la elipse.

La clave de este algoritmo es determinar que puntos pertenecen al objeto antes de desplazar. Para esto se puede establecer lo siguiente: todo punto perteneciente al objeto va a estar entre cuatro puntos a menos que sea el borde de este. Por esta razón, el algoritmo parte recorriendo toda la matriz imagen hasta detectar un punto objeto. Una vez localizado se calcula el ángulo relacionado, el cual siempre va a estar entre 0 y 90 grados, ya que se calcula en base a la distancia entre los puntos y el cateto opuesto, por lo que, determinando en qué parte del plano está en relación al punto (h, k) se le suma lo faltante. La Fig. 19 recrea esta situación. Como el punto $(x1, y1)$ a proyectar se encuentra en la esquina superior

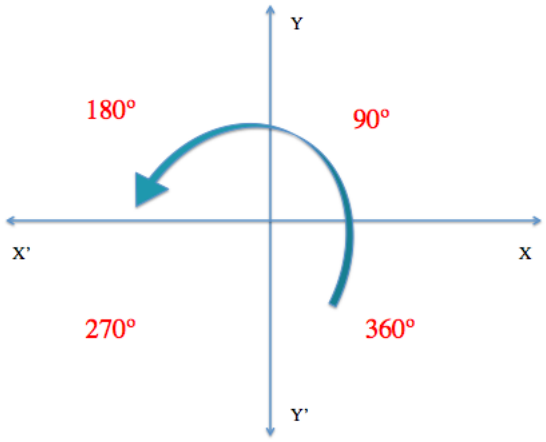

Figura 17: Medición de grados en un plano cartesiano con centro en (0,0).

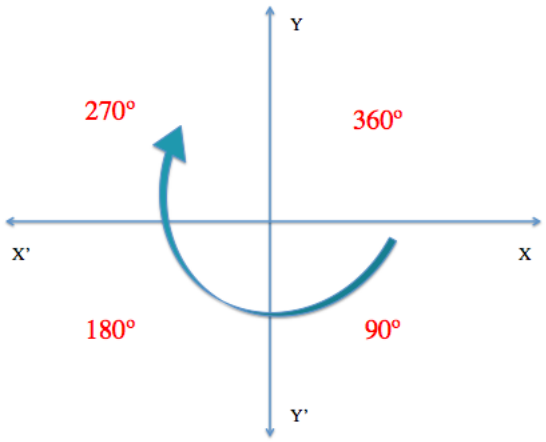

Figura 18: Medición de grados en un arreglo bidimensional con centro en $(n/2, m/2)$.

Algorithm 9 Algoritmo de deproyección.

```
1: Función deproyectar(x, y, NAXIS1, NAXIS2, borde, matriz, radio):
2: while i != NAXIS1 do
3:   while j != NAXIS2 do
4:     a = b = c = d = 0
5:     while r != largo(borde - 1) do
6:       if (r es impar) then
7:         continúe
8:       end if
9:       if i < borde[r] then
10:        a = 1
11:      end if
12:      if i > borde[r] then
13:        b = 10
14:      end if
15:      if j > borde[r + 1] then
16:        c = 100
17:      end if
18:      if j < borde[r + 1] then
19:        d = 1000
20:      end if
21:      if i == borde[r] AND j == borde[r + 1] then
22:        a, b, c, d = 1,10,100,1000
23:        break
24:      end if
25:    end while
26:    if (a + b + c + d != 1111 OR x == i AND y == j) then
27:      continúe
28:    end if
29:    ángulo = arcoseno((j − y)/distancia(x, y, i, j))
30:    while TRUE do
31:      if i > x AND j > y then
32:        ángulo = ángulo
33:        break
34:      end if
35:      if i < x AND j > y then
36:        ángulo = 360 – ángulo
37:        break
38:      end if
39:      if i < x AND j < y then
40:        ángulo = 180 + ángulo
41:        break
42:      end if
43:      if i > x AND j < y then
```

```
44:            ángulo = 180 – ángulo
45:            break
46:          end if
47:          if i == x AND j > y then
48:            ángulo = 0
49:            break
50:          end if
51:          if i == x AND j < y then
52:            ángulo = 180
53:            break
54:          end if
55:          if i > x AND j == y then
56:            ángulo = 90
57:            break
58:          end if
59:          if i < x AND j == y then
60:            ángulo = 270
61:            break
62:          end if
63:        end while
64:      end while
65:      fil = x + (seno(ángulo * 3.14/180)) * distancia
66:      col = y + (coseno(ángulo * 3.14/180)) * distancia
67:  end while
68:  matriz_final[fil,col] = matriz[i][j]
69:  retornar matriz_final
```

derecha del centro y dado que los ángulos se cuentan en sentido antihorario, el ángulo no vale $\partial°$, sino más bien 360 - ∂. Si hubiésemos tomado un punto de la esquina superior izquierda, su ángulo sería 180 + ∂, si el punto está en la parte inferior izquierda 180 - ∂, y finalmente si ubiese estado en el lado inferior derecha, el ángulo se mantiene. Ya determinado el ángulo, el algoritmo ocupando la ecuación paramétrica de un punto en el círculo, y calculando brevemente la distancia que debería estar el punto en relación al punto más distante, se obtiene la coordenada del punto proyectado. Este proceso se repite tantas veces como puntos pertenezcan al objeto de estudio. Una vez finalizado todo el proceso, se retorna una matriz que posee la proyección de la elipse a un círculo.

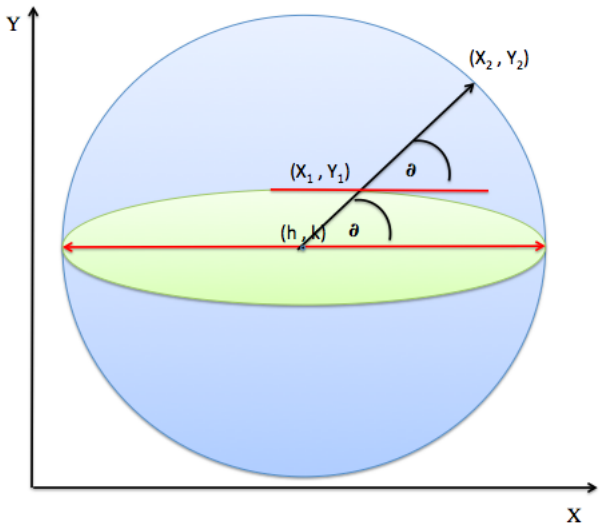

Figura 19: Proyección de un punto elíptico a un punto del círculo.

2.4. Funciones especiales

Si bien el registro de imágenes contempla cuatro niveles de ajustes, detrás de estos módulos actúan constantemente funciones que entregan información valiosa antes de realizar cualquier ajuste a un objeto en el plano. Estas funciones son 'invisibles' para el usuario, es decir, no entregan resultados tangible u observables, sino más bien es el sistema de cómputo que aporta información sobre los cuerpos geométricos, como tamaño, grado de inclinación y posición, para que los módulos que realizan el registro sepan a priori en qué medida ajustar el objeto.

Distancia entre puntos Para un objeto elíptico posicionado dentro de una imagen es necesario medirlo antes de escalarlo, de esta manera se puede determinar que tan grande o más pequeño es el objeto de la imagen actual, con la imagen de referencia. La medición de un objeto se basa en determinar la distancia entre los puntos más distantes pertenecientes al borde de la elipse, conocida también como distancia euclidiana, la cual se calcula mediante la Ec. 13.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{13}$$

La Ec. 13 indica que la raíz cuadrada de la suma de la diferencia de las coordenadas al cuadrado es igual a la distancia d. Algorítmicamente se ve de como en el Alg. 10.

Algorithm 10 Algoritmo de distancia Euclidiana.

1: Función distancia (x1, y1, x2, y2):
2: retornar raiz_cuadrada($(x2 - x1)^2 + (y2 - y1)^2$))

La función recibe como parámetros de entrada el par de coordenadas de las cuales se desea saber su distancia, para luego al igual que la fórmula anterior retornar la distancia.

Distancia eje mayor Esta función no es más que una aplicación de la función distancia, lo distinto es que busca la mayor distancia entre todos los puntos que componen el borde del objeto elíptico. De esta manera cuando encuentra los puntos más distantes, sabe de inmediato que corresponde al eje mayor de la elipse.

La función del Alg. 11, recibe como parámetro de entrada un arreglo que contiene todas las posiciones del borde de la elipse, de las cuales se localizará las más distantes.

En la línea número dos, se crea un vector que guardará en su posición 0 la distancia del eje, y en las posiciones restantes sus coordenadas, por ejemplo, si el eje mayor mide 30 unidades, y sus coordenadas son $(X1, Y1)$ y $(X2, Y2)$, el vector se escribe como en la Ec. 14.

$$\text{info}_{\text{vector}} = [30, X_1, Y_1, X_2, Y_2] \tag{14}$$

Este vector es iniciado con ceros, ya que cada posición actúa como una variable auxiliar mientras se busca el eje mayor.

Los ciclos **while** de la línea 3 y 8 permitirán encontrar la mayor distancia, para esto el ciclo de la línea 3 va seleccionando una coordenada del vector y el ciclo de la línea 8 compara la coordenada elegida en el ciclo 3 con todas las coordenadas restantes. Si esta resulta ser mayor que la encontrada anteriormente, se reemplaza en la posición cero del vector 'info_vector' la distancia encontrada y se actualizan las coordenadas de las cuales se extrajo la distancia.

Ángulo a rotar (Alg. 12) Para rotar un objeto en el plano es necesario saber cuánto es lo que se necesita rotar. La siguiente función determina a partir de la información generada anteriormente cuánto es lo que necesita rotarse el objeto para que su eje mayor quede paralelo al eje horizontal en un plano.

La función recibe como entrada las coordenadas extremas del eje mayor del objeto. En la línea dos pregunta si el eje mayor es paralelo al eje y, de esta manera se debe rotar, de lo contrario calcula el arcotangente del ángulo. En la línea cinco, se le resta esta cantidad a 180, esto es debido a que la función

Algorithm 11 Algoritmo de distancia al eje mayor

```
1: Función eje_mayor(borde):
2: info_elipse = [0,0,0,0,0]
3: while i distinto largo(borde) - 2 do
4:    if (i es impar) then
5:       continúe
6:    end if
7:    i = i + 2
8:    while j distinto largo(borde) - 1 do
9:       if (j es impar) then
10:         continúe
11:      end if
12:      dis = distancia(borde[i], borde[i + 1], borde[j], borde[j + 1])
13:      if dis > info_elipse[0] then
14:         info_elipse[0] = dis
15:         info_elipse[1] = borde[i]
16:         info_elipse[2] = borde[i + 1]
17:         info_elipse[3] = borde[j]
18:         info_elipse[4] = borde[j + 1]
19:      end if
20:   end while
21: end while
22: retornar info_elipse
```

Algorithm 12 Algoritmo de ángulo a rotar

```
1: Función angulo_a_rotar (x1, y1, x2, y2):
2: if (y2 − y1) == 0 then
3:    retorne 90
4: else
5:    retorne 180 – arcotangente ((y2 − y1)/(x2 − x1))
6: end if
```

que realiza la rotación gira en sentido antihorario el cuerpo, por tal motivo, cuando se determina el ángulo de la recta se calcula su complemento.

Matemáticamente consiste en determinar la tangente del ángulo ∂ que forma la recta con la dirección positiva del eje de las abscisas, por lo tanto, para determinar el ángulo se realiza el proceso inverso (Ec. 15 y Ec. 16).

$$m = \frac{y_2 - y_1}{x_2 - x_1} \tag{15}$$

$$\partial = \arctan m \tag{16}$$

Centro geométrico (Alg. 13) Al momento de alinear un objeto es necesario determinar su punto central para poder trasladar esa coordenada al punto que se tiene como referencia. Para poder determinar la coordenada del centro geométrico del objeto, se calcula a partir del tamaño de la mitad del eje mayor de la elipse, la proyección del ángulo en el eje horizontal y vertical.

Algorithm 13 Algoritmo para calcular centro geométrico

```
1: Función centro_geométrico(dis,x1,y1,x2,y2,angulo):
2: if (ángulo < 90) then
3:    punto_central = [x1 + 0.5 * coseno(ángulo) * dis, y1 + 0.5 * seno(ángulo)*dis]
4: end if
5: if (ángulo > 90) then
6:    punto_central = [x2 - 0.5 * cos(angulo) * dis, y2 – 0.5 * seno(ángulo) * dis]
7: end if
8: if (ángulo == 90) then
9:    punto_central = [(x1 − x2)/2 , y1]
10: end if
11: retornar punto_central
```

La función recibe como parámetro de entrada la medida del eje 'dis', las coordenadas extremas y el ángulo que corresponde a la pendiente.

La clave de este algoritmo es saber si la pendiente del eje es positiva o negativa, ya que la proyección de la recta en los ejes, se hace a partir de la coordenada más baja del objeto al punto central, por este motivo, se pregunta si el ángulo de la recta es mayor o menor que 90°. Al obtener el punto central del eje, debe sumarse la coordenada menor del eje del objeto, esto se hace de esta manera ya que el punto centro encontrado es el centro del objeto y no el centro del objeto en relación al plano. Por tal motivo en la linea tres y cinco del plano se suma $(x1, y1)$ y $(x2, y2)$. Finalmente, se retorna la coordenada.

Estos cuatro últimos algoritmos entregan toda la información necesaria para que cada etapa del registro realice correctamente un ajuste de la información. Partiendo desde la toma de medidas del objeto de referencia, hasta la actualización del mismo en cada etapa.

3. Conclusiones

A lo largo de este documento se ha visto como se aplica el registro de imágenes a objetos astronómicos y la utilidad junto con el valor que genera su uso en investigaciones en diferentes áreas mas allá de las ciencias del universo.

Se puede apreciar la importancia que ha tenido el registro en el ámbito de la medicina nuclear, ayudando a observar, caracterizar y definir alteraciones físicas a nivel cerebral que presentan pacientes, permitiendo comparar resultados en el tiempo y así definir que tratamiento es el mas adecuado.

También se habla de un proyecto llamado CARTOSUR donde se mezclaron imágenes en diferentes longitudes de onda. Aquí se aprecia que si bien el registro de imágenes es una técnica que remota desde los inicios de la imagen digital, el incursionar en nuevas técnicas, como mezclar imágenes de radio con imágenes ópticas, es la prueba de lo amplio y extenso que puede ser el uso del registro.

La necesidad existente en poseer mejores herramientas de análisis que apoyen la toma de decisiones es lo que motiva la exploración, experimentación y desarrollo de técnicas que procesen la información.

Atacama Large Milimetric/Submilimetric Array, actualmente el radio telescopio más grande del mundo, genera tanta información, alrededor de 750Gb de información por día, que los astrónomos requieren contar con una plataforma abierta para acceder virtualmente a todas las observaciones realizadas. Esta plataforma virtual nombrada 'ChiVO', necesita contar con herramientas que procesen de manera inteligente los datos capturados, generando información de calidad, útil para los astrónomos y comunidad en general. En el caso del registro de imágenes, es una herramienta avanzada de mucha utilidad tanto para ALMA como otros observatorios, dando un valor agregado a la información obtenida. Esto es de vital importancia, ya que el registro no es tan solo disminuir el ruido o aumentar la calidad de una imagen, sino las variadas posibilidades de análisis que permite realizar, por ejemplo: estudiar la composición o estructura de una galaxia en particular a partir de la información obtenida en una observación; determinar la edad que posee un cúmulo globular; analizar la evolución o cambios que ha tenido un cuerpo en el tiempo. En general, los algoritmos implementados a lo largo de las etapas del registro de imágenes son operaciones sencillas sobre matrices y transformaciones geométricas en el plano, por lo que en la bibliografía existente no hay mucha variación. El registro básicamente es una aplicación que hace corresponder a cada punto P de coordenadas (x, y) del plano, otro punto P' de coordenadas (x', y') del mismo plano. Si bien la distribución de coordenadas en el plano cartesiano no poseen exactamente la misma estructura en que se lee una matriz computacionalmente, la geometría aplicada varía levemente, solo hay que tener presente que lo que habitualmente leemos en un plano cartesiano como abscisa y luego ordenada, aquí es al revés, se lee como fila columna, y los ángulo formados en los ejes van en sentido horario. Asumiendo lo anterior, ya es mas sencillo trabajar con la información.

Otro aspecto fundamental en el registro de imágenes, es la forma en que completamos la información cuando se realiza una modificación, una matriz computacional no posee puntos densos, es decir, al subdividir el espacio entre dos coordenadas, no necesariamente existirá otra coordenada entera y por otro lado al trasladar la información existente, puede que queden puntos sin información. En estos casos la interpolación es imprescindible para completar la información faltante. Si bien cada autor tiene su forma, la idea base es completar a partir de los puntos vecinos el punto que esta ausente, algunos ocupan los 4 vecinos mas cercanos, otros 16, pero depende del fin que se quiere conseguir. En el caso de los algoritmos

implementados en este trabajo tienen una mirada distinta. Por un lado, cuando ocurren situaciones en que la información cae sobre un punto 'decimal', por ejemplo (2.4, 3), el sistema decide dejar la información repartida entre el punto (2.0, 3) y (3.0, 3). Este esquema es mucho mas sencillo, por un lado asegura que todos los puntos quedarán uniformemente distribuidos y por otro lado no es invasivo, es un trabajo mas sutil. El segundo caso ocurre cuando una imagen es ampliada, la cantidad de información vacía intermedia que queda, es proporcional a la razón de escala utilizada. Mirar los objetos mas cercanos para completar el punto faltante es algo tedioso, ya que desde un punto vacío los vecinos no están equidistantes, dada esta situación los vecinos colaboran para completar lo que entre ellos falta, esto es mucho mas sencillo, ya que desde un punto con información a otro punto con información la distancia es la razón. La cantidad de vecinos a comunicar son en grupos de cuatro, de los cuales, dos de ellos se ocuparán para el conformar el grupo siguiente.

Un aspecto importante dentro del proceso de ajuste, es contar con el conjunto extra de funciones que entregan la información necesaria para que el ajuste tome valor, es decir, ya no tan solo son herramientas que extraen, rotan, escalan, alinean o deproyectan un objeto, sino que aportan la información necesaria para determinar en que grado conviene rotar, en que punto alinear, que tanto agrandar o achicar, o donde desplazar los puntos para que queden deproyectado. Con esto ya se marca una diferencia más a una herramienta de registro convencional.

A lo largo de las pruebas realizadas para medir la eficacia de los algoritmos, se puede decir, que fueron satisfactoriamente logradas, pero no son lo mas óptimo al momento de operar en un computador de escritorio. El tiempo requerido para realizar algunas operaciones como la rotación de un objeto, registro una duración de alrededor de 20 minutos sobre un conjunto de 100 imágenes de 1095 x 1095 pixeles, sin contar el tiempo que requeriría para terminar con el resto.

Si bien el objetivo de la tesis de realizar un módulo en Python que realice el registro de imágenes astronómicas se cumple, es el puntapié inicial para mejorar, optimizar y desarrollar todo lo realizado en una plataforma virtual como es el Observatorio Virtual Chileno. Adicionalmente se puede expandir las funcionalidades, y realizar aplicaciones en otros tipos de objetos extrasolares, como por ejemplo, nebulosas, discos de acreción, exoplanetas, etcétera, como también trabajar con imágenes en otras dimensiones, por ejemplo, líneas espectrales o cubos de información. La idea es apuntar a crear o mejorar herramientas que potencien aún mas la información, colaborando en el desarrollo científico y así permitir a los investigadores invertir su tiempo en analizar o estudiar la información, mas que en como procesarla.

Lo mas hermoso de todo lo realizado, es ver como un código toma vida y se plasma como un sentimiento en la tarea de un astrónomo que busca cada día descubrir y conocer lo que hay mas allá de sus ojos.

A. Galaxias

Todo sistema independiente de estrellas situado fuero de la Vía Láctea, se le atribuye el nombre de galaxia. Las galaxias que poseen un diámetro de entre 6.000 a 170.000 años luz, contienen entre 3.000 millones a 1 billón de estrellas, así como una cantidad de gas y polvo que representa el 10 % de su masa total.

Las galaxias se clasifican, atendiendo a su morfología, en tres grandes grupos:

a Galaxias elípticas: Este conjunto de galaxias representan el 20 % de las galaxias conocidas. Poseen forma esferoidal y carecen por completo de estructura interna definida. Además, no contienen apenas materia interestelar y las estrellas que la componen son viejas y se encuentran en estados muy avanzados de evolución.

b Galaxias espirales: Estas están divididas en dos grupos, el de las espirales normales y el de las espirales barradas. Las primeras, que representan el 75 % de todas las galaxias, tienen forma de lente aplanada y están dotadas de 2 a 3 brazos que parten de su centro, así como de gran cantidad de materia interestelar y muchas estrellas brillantes y jóvenes. La Vía Láctea pertenece este tipo. Por otro lado, las espirales barradas, disponen de dos brazos rectos opuestos que parten de su centro y de cuyos extremos surgen casi perpendicularmente sus brazos espirales, que en algunos casos se desarrollan hasta formar un círculo alrededor del núcleo.

c Galaxias irregulares: Representando el 5 % del total, su forma no presenta simetría alguna, siendo su aspecto y forma muy variables. Por lo general son pequeñas y contienen gran cantidad de materia interestelar. Se subdividen en las de tipo I, que pueden resolverse en estrellas, nebulosas o cúmulos, y las de tipo II, que no admiten este tipo de resolución y parecen ser completamente amorfas.

B. Apilado de imágenes. Aplicación post registro

Las imágenes astronómicas obtenidas por cualquier telescopio se ven limitadas por varios factores. Entre ellos, las condiciones atmosféricas, la resolución del telescopio o la razón señal/ruido en cada pixel de la imagen. La señal captada en cada pixel, depende del flujo de la fuente astronómica o el tiempo de integración. El ruido existente en cada pixel depende netamente del instrumento de observación. Por ejemplo, en un circuito de carga acoplada (CCD), el ruido se genera por la agitación térmica de los portadores de carga (conductores), por la lectura del CCD y el 'fondo' de emisión del cielo y/o telescopio. En general, un mayor tiempo de integración en una fuente óptica proporciona imágenes con una señal mas expuesta al ruido.

En la obtención de imágenes de radio el ruido se genera principalmente por la agitación térmica de los receptores ubicados en cada telescopio. Aquí, interacciones más largas dan lugar a imágenes con mejores niveles de señal respecto al ruido, debido a que la integración ya produce una disminución. Para un telescopio como ALMA, la relación señal/ruido es inversamente proporcional a la raíz cuadrada del producto del ancho de banda y el tiempo de integración total. Es decir, para un ancho de banda fijo, la duplicación de la señal al ruido requiere cuatro veces el tiempo de integración.

En el caso del observatorio ALMA, en muchos casos, no se detectan objetos astronómicos individuales en una imagen, es decir, la señal de éste objetivo es más pequeño que el ruido (o tres veces el ruido) en la imagen. Sin embargo, si uno tiene una muestra de los objetivos (por ejemplo, 1.000 fuentes) con posiciones conocidas con precisión, los cuales están en la imagen, se podrían extraer sub-imágenes centradas en cada posición y la media de todas las 1000 sub-imágenes. La imagen promedio es equivalente a una imagen con 1000 veces el tiempo de integración, es decir, el ruido en la imagen se reduce por un factor de la raíz cuadrada de 1000 o aproximadamente 30. Por tanto, es más probable que la imagen promediada muestre una detección de señales muy débiles que requerirán de varios días o meses de observación (la señal en la imagen promediada es la señal media de las 1000 fuentes, mientras que el ruido es 30 veces menor que el ruido en cualquiera de las imágenes en las 1000 fuentes). Ésta técnica, conocida como apilamiento, es una herramienta muy potente y se utiliza por ejemplo, en imágenes de rayos X, formación de imágenes ópticas e imágenes de radio. El apilamiento da una idea de las características medias de una muestra, algo muy útil para el análisis.

Como el archivo de ALMA sigue creciendo, una fracción cada vez mayor del cielo está siendo capturada. Con una muestra suficientemente amplia de objetos astronómicos, por ejemplo, un millón de cuásares con posiciones precisas del Sloan Digital Sky Survey, SDSS, se encuentra que muchos objetos de la muestra estarán en áreas que ya han sido observadas por ALMA. Incluso, si no se detectan de forma individual, se puede apilar todos los objetos que han sido vistos por ALMA y obtener un flujo promedio para ésta muestra. Si la muestra también tiene desplazamientos al rojo, el apilamiento se puede realizar en el espacio de velocidades con el fin de obtener las propiedades medias de cualquier línea de emisión dada.

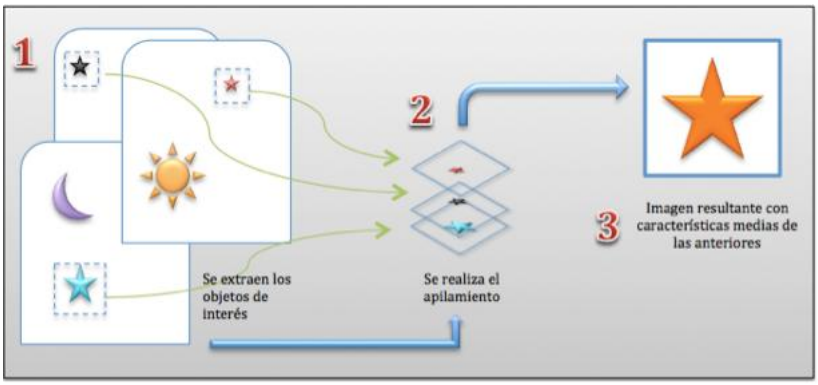

Figura 20: Proceso de apilado de imágenes a partir de un conjunto de imágenes normalizadas.

C. Inicios del apilado de imágenes

Todas las observaciones de imágenes astronómicas poseen un umbral de sensibilidad por debajo del cual los objetos no son detectables. Si uno tiene razones para creer que en esa posición del cielo puede existir algo, basándose en otras observaciones, se puede aplicar apilamiento a una zona delimitada a un cierto rango de longitudes de onda, pudiendo variar dicha longitud, y así conseguir el flujo medio de emisiones.

En una aplicación inicial de esta técnica, Caillault y Helfand, detectaron el flujo medio de rayos X de estrellas G en Pleiades, utilizándola para determinar la edad respecto al deterioro de la emisión. Los detectores de emisiones de rayos X han estado disponible durante más de dos décadas, por lo que el apilamiento se ha convertido en una técnica de análisis estándar. Las aplicaciones comenzaron desde la determinación de la luminosidad media de rayos X, por ejemplo las galaxias normales, galaxias Lyman y las fuentes de radio, para determinar la emisión de rayos X de grupos distantes en el Röntgensatellit (ROSAT) All-Sky Survey.

Los detectores digitales lineales han llegado a dominar los estudios del cielo óptico e infrarrojo. La técnica de apilamiento ha sido ampliamente adoptada: por ejemplo, Zibetti et al. detectaron la luz intracluster apilando Sloan Digital Sky Estudio 683 (SDSS6); Lin et al., apiló datos sobre las galaxias del cúmulo Two Micron All Sky Survey (2MASS); Hogg et al. apiló datos del Keck IR para conseguir galaxias de colores tenues; Minchin et al. fue tan lejos como para apilar películas digitalizadas desde el telescopio Schmidt del Reino Unido.

Referencias

[ALM] ALMA. El universo del ALMA. Santiago, Chile.

[Ast] Astropy. Edit a FITS header. Retrieved 12 de 07 de 2013 from Astropy Tutorials: `http://www.astropy.org/astropy-tutorials/FITS-header.html`.

[EXP09] EXPLORA CONICYT. *Noticias del universo*, number 21 in Expoastronomía, 2009.

[Gar] G. Garay. Introduction to radio-interferometry. Technical report, Departamento de Astronomía, Universidad de Chile, Santiago, Chile.

[Gre12] E. W. Greisen. AIPS FITS File Format, 2012.

[Hei07] M. Heiseg. Cerca del cielo. Technical report, ESO, Santiago, 2007.

[Ins00] Instituto de Astrofísica de Andalucía. *Diez años de investigación con el telescopio espacial HUBBLE*, number 20 in Información y Actualidad Astronómica, 2000.

[J.95] L. A. J. Fusión de imágenes del satélite landsat tematic mapper y de radar de apertura sintética aerotransportado para facilitar la delineación del litoral. Technical report, Louisiana State University, USA, 1995.

[LSS+00] P. J. Lewis, A. Siegel, A. M. Siegel, C. Studholme, Sojko-va, J., and D. W. Roberts et. al. Does performing image registration and subtraction in ictal brain spect help localize neocortical seizures? Technical report, Department of Radiology, USA, 2000.

[Mig03] J. Miguel. Integración de ortoimágenes de radar del proyecto cartosur con datos ópticos provenientes de las imágenes landsat con fines de interpretación visual. Technical report, Instituto Geográfico de Venezuela Simón Bolívar, Venezuela, 2003.

[Pol] Fundación Polar. Matrices y transformaciones.

[Pyt] Python. Mathematical functions. Retrieved 12 de 05 de 2013 from Numeric and Mathematical Modules: `https://docs.python.org/2/library/math.html`.

[RM12] J. Rogan and V. Muñoz. Programación y métodos numéricos. Technical report, Universidad de Chile, Chile, 2012.

[Ter13] La Tercera. Datos de alma serán parte de la alianza internacional de observatorios virtuales, 12 de mayo 2013.