-A. Descripción global del método

La ejecución del método de ejemplares (en todas sus variantes) consiste en una secuencia de pasos a ejecutar de manera iterativa para reconstruir la zona U desconocida de la imagen. Estos pasos son tres

- Cálculo del borde
- Selección del punto del borde a eliminar
- Selección de rectángulo correspondiente para el reemplazo

En general las distintas lineas de investigación buscan optimizar cada una de estas etapas para lograr por un lado una mayor efectividad, y por el otro una mayor eficiencia.

-A1. Cáclulo del borde: Es indispensable para el funcionamiento del algoritmo calcular el borde de la región U a eliminar, ya que el algoritmo siempre copia un ejemplar del exterior al borde del interior de la región U. Hay diversos algoritmos que cálculan bordes, forman parte del area de la geometria computacional, el Convex Hull (wikipedia: https://en.wikipedia.org/wiki/Convex_hull) por ejemplo es un algoritmo que aplica para resolver el problema. Generalmente al ser un problema conocido existen infinidad de implementaciones ya realizadas de codigo libre para resolver este problema

-A2. Cálculo del gradiente de la imagen: Para el funcionamiento del algoritmo fue necesario el cálculo de los gradientes de la imagen. Para ello se cálculo primero la imagen en una escala de grises y luego se aplicó el operador Sobel, el cual combinó tanto derivada como suavización gaussiana.

-A3. Selección del punto del borde a eliminar: En cada iteración se necesita calcular de todos los pixeles del borde aquellos con una mayor prioridad para ser asimilados. En los papers citados se muestran fundamentos por los cuales es más apropiado poderar por un lado el gradiente de la imagen y por el otro la confianza La formula que determina la prioridad esta

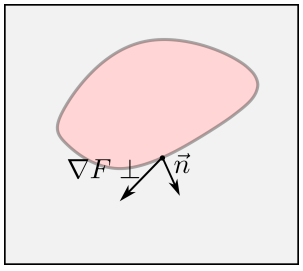

Figura 1. Gradiente y normal

dada por

$$P_i = C_i \nabla F_i \perp \vec{n}_i \tag{1}$$

Donde $\nabla F_i \perp$ es la perpendicular al gradiente de la imagen en el pixel i. Es el punto donde ocurre el cambio más suave en la intensidad del color de la imagen. El algoritmo realiza el producto escalar entre $\nabla F_i \perp \vec{n}_i$ lo cual consigue la componente normal del la perpendicular al gradiente, que nos indica que tan suave cambia la intensidad de la imagen en la dirección normal al borde de la región

-B. Determinación del rectángulo a reemplazar

Se necesita calcular cual réctangulo utilizar para reemplazar la zona del borde de la imagen. Para ello se debe desarrollar un criterio que determine cuál es el más apto. El metodo más sencillo consiste en realizar la norma de la diferencia comparando la región a reemplazar y el candidato. Se utiliza la siguiente ecuación.

$$\sum (R_{1i} - R_{2i})^2 + (G_{1i} - G_{2i})^2 + (B_{1i} - B_{2i})^2 \tag{2}$$

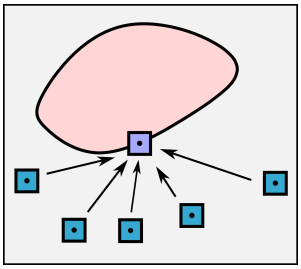

Figura 2. Descripción del rectangulo a reemplazar

Lamentablemente este método tiene el incoveniente de ser muy lento ya que exige una comparación de pixel a pixel para cada candidato. Si bien una selección de los candidatos a comparar con azar involucrado puede tener buenos resultados, se procedió a estudiar un método, patch Match que mejore este aspecto puntual del algoritmo.

-B1. Optimización mediante PatchMatch: Un enfoque por fuerza bruta de la selección del rectangulo correspondiente para el reemplazo puede resultar excesivamente costoso en tiempo, ya que implicaría potencialmente comparar una región de la imagen con todas las demás. Por lo tanto, existen diversos criterios que pueden tomarse en este sentido.

Uno de ellos es desarrollado por Barnes, Shechtman, Finkelstein y Goldman, quienes desarrollaron un algoritmo con el nombre de PatchMatch [**?**]. El mismo hace uso de la asunción de ciertas propiedades de coherencia en la imagen para reducir el número de comparaciones que debe hacerse, sin perder la efectividad del algoritmo.

Una vez elegido un lugar donde se quiere rellenar, PatchMatch comienza comparando con algunas regiones aleatorias de la imagen original, y a partir del resultado más satisfactorio que obtiene hace dos suposiciones:

- si se tiene un buen match para una región que se quiere rellenar, es probable que, si existe un parche que es un match aún mejor, se encuentre cerca del bueno (es decir, se asume que estamos en la zona correcta de la imagen, y sólo debemos terminar de acercarnos al match óptimo)
- si un parche es un buen match para una cierta región a rellenar, es probable que también sea un buen match para regiones cercanas

De esta manera, según los autores del algoritmo, se logra obtener muy buenos resultados con menos comparaciones. La limitación que tiene este algoritmo es que no puede dejarse de

introducir comparaciones random incluso habiendo encontrado un “buen” match, puesto que se corre el riesgo de quedarse en un mínimo local, que no sea el mínimo global.

-B2. Pruebas del método: Se procedió a ejecutar el método con algunos casos simples:

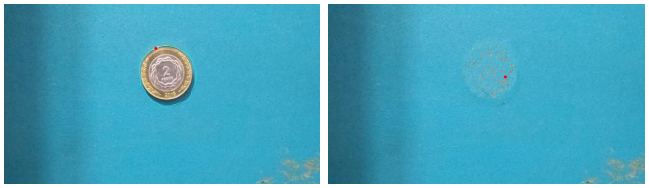

Figura 3. Resultados - ejemplo 1

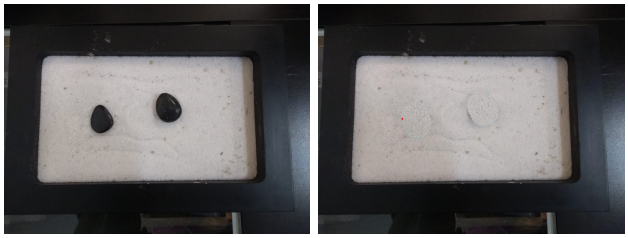

Figura 4. Resultados - ejemplo 2

Estas son algunas pruebas iniciales, las cuales mostraron algunos desperfectos en el método original, como por ejemplo que la región reemplazada se vea ”suciaz también dieron una noción de la tardanza del algoritmo.