22.90 – Automación Industrial

Visión: Funciones morfológicas, funciones de forma, y momentos

Ing. Rodolfo Arias

Objetivo

En esta clase, revisaremos los conceptos de operaciones morfológicas, warping genérico y momentos de la imagen.

Utilizaremos las operaciones morfológicas como filtro basado en 'formas' dentro de la imagen.

Veremos algunas funciones básicas de modificación de tamaño, posición y rotación de la imágenes (englobados en técnicas de warping)

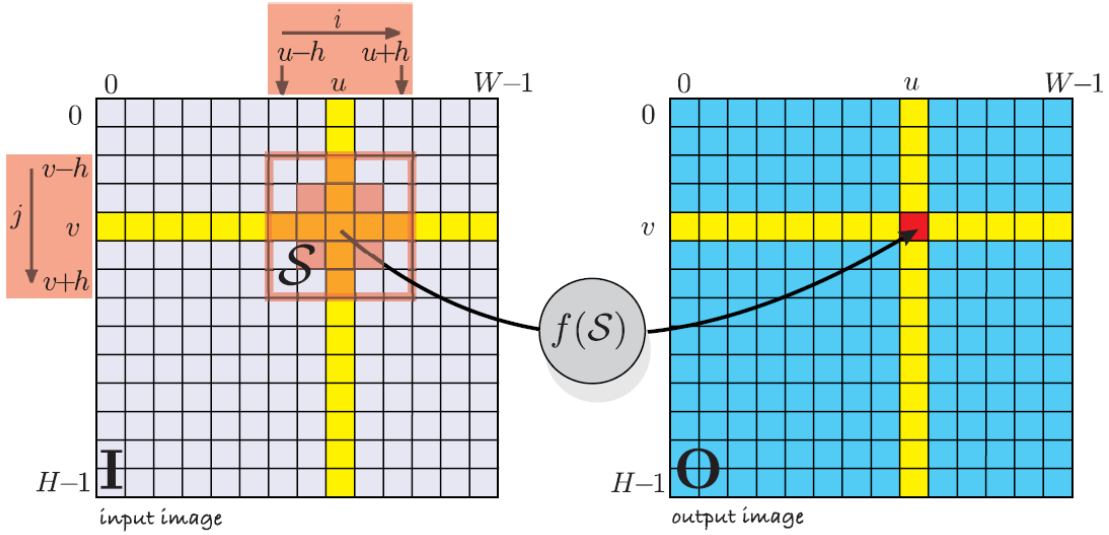
Por ultimo, revisaremos el concepto de momentos de una imagen, la forma de calculo, y el sentido 'físico' de los mismos.

Morfología

Operador espacial no lineal. Matemáticamente el concepto puede resumirse como:

$$\mathbf{O}[u,v] = f\big(\mathbf{I}[u+i, v+j]\big), \quad \forall (i,j) \in \mathcal{S}, \quad \forall (u,v) \in \mathbf{I}$$

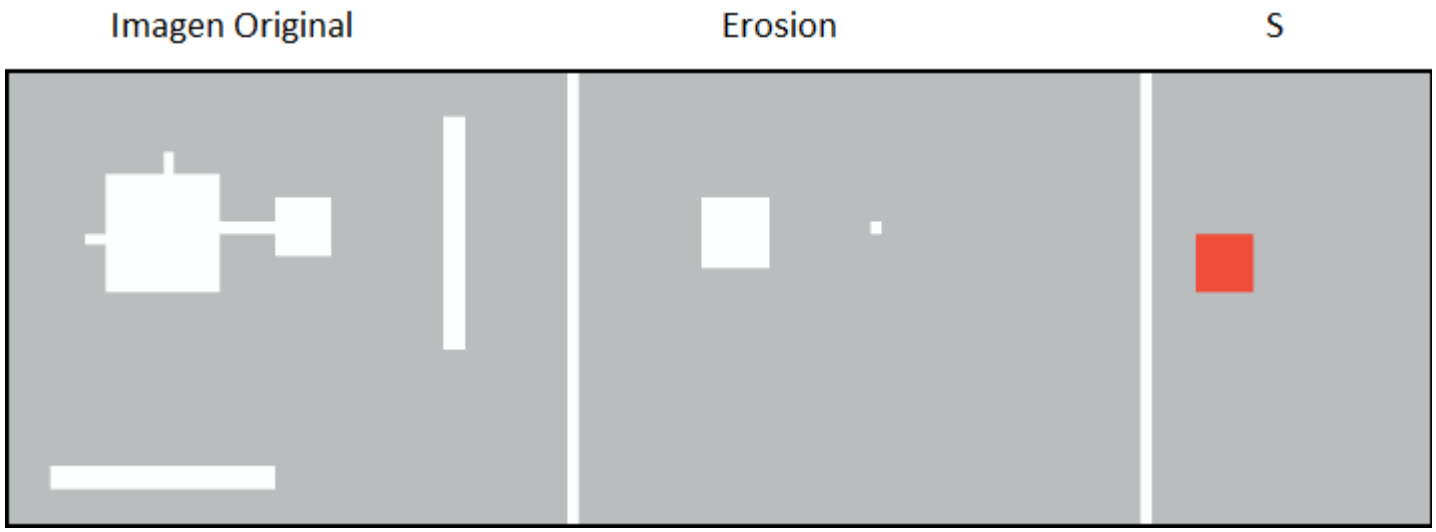
Donde S es el elemento estructural.



Erosión

- La erosión es una de las operaciones morfológicas principales. La misma busca el valor mínimo presente en la figura estructural S, centrada en el pixel de interés.

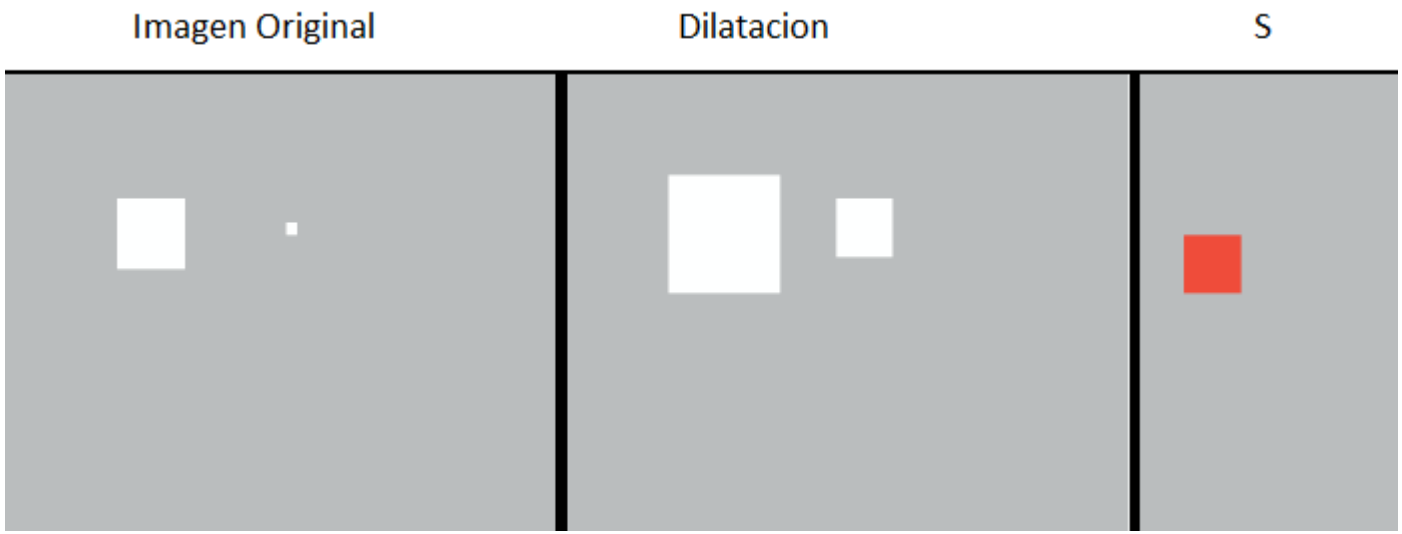
```
>> S = ones(5,5);
>> mn = imorph(im, S, 'min');= ierode(im, S);
```



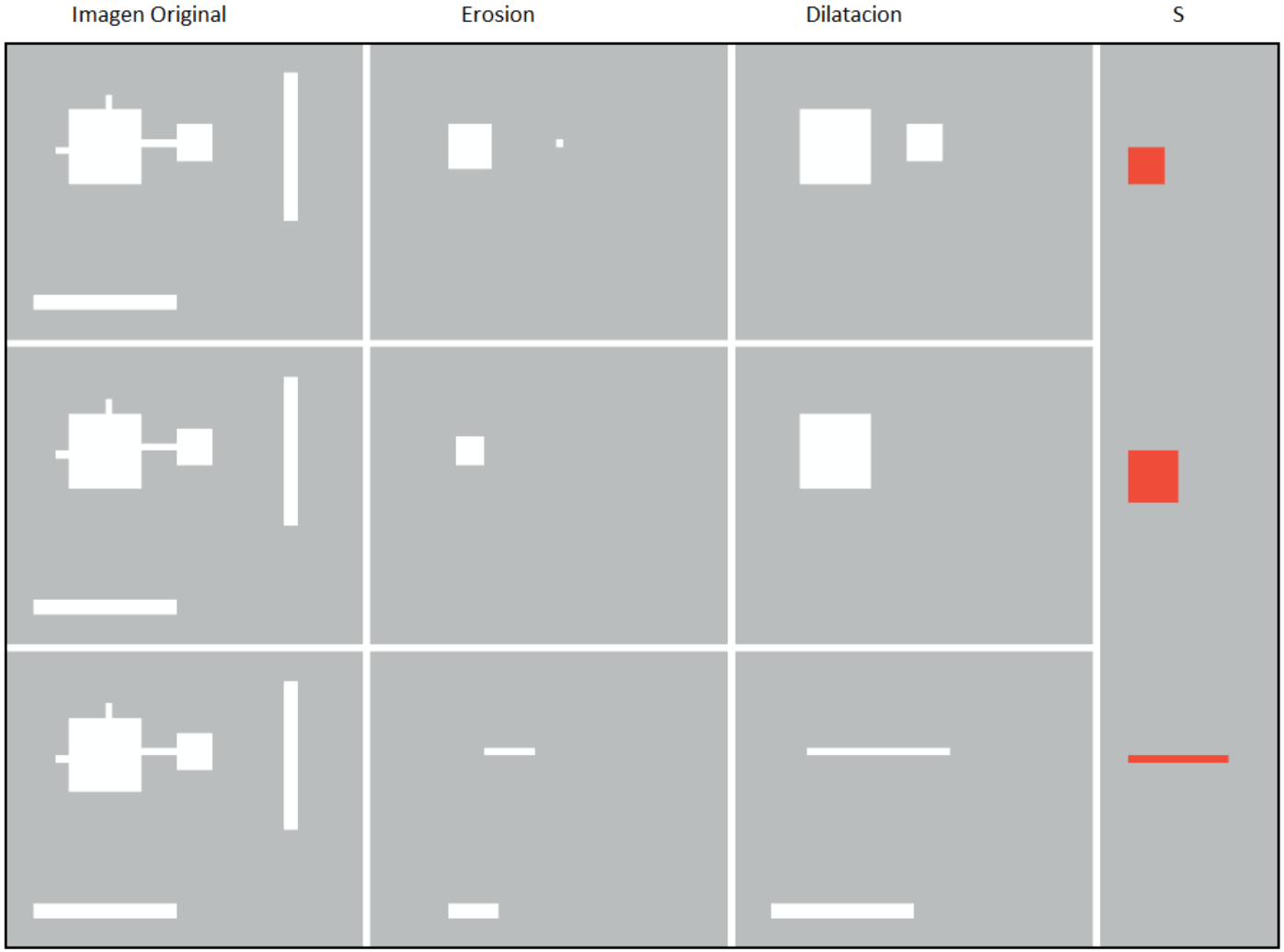
Dilatación

- Operación opuesta a la erosión, esta función busca el valor máximo en el elemento estructural S.

```
>> mx = imorph(mn, S, 'max');= idilate(im, S);
```

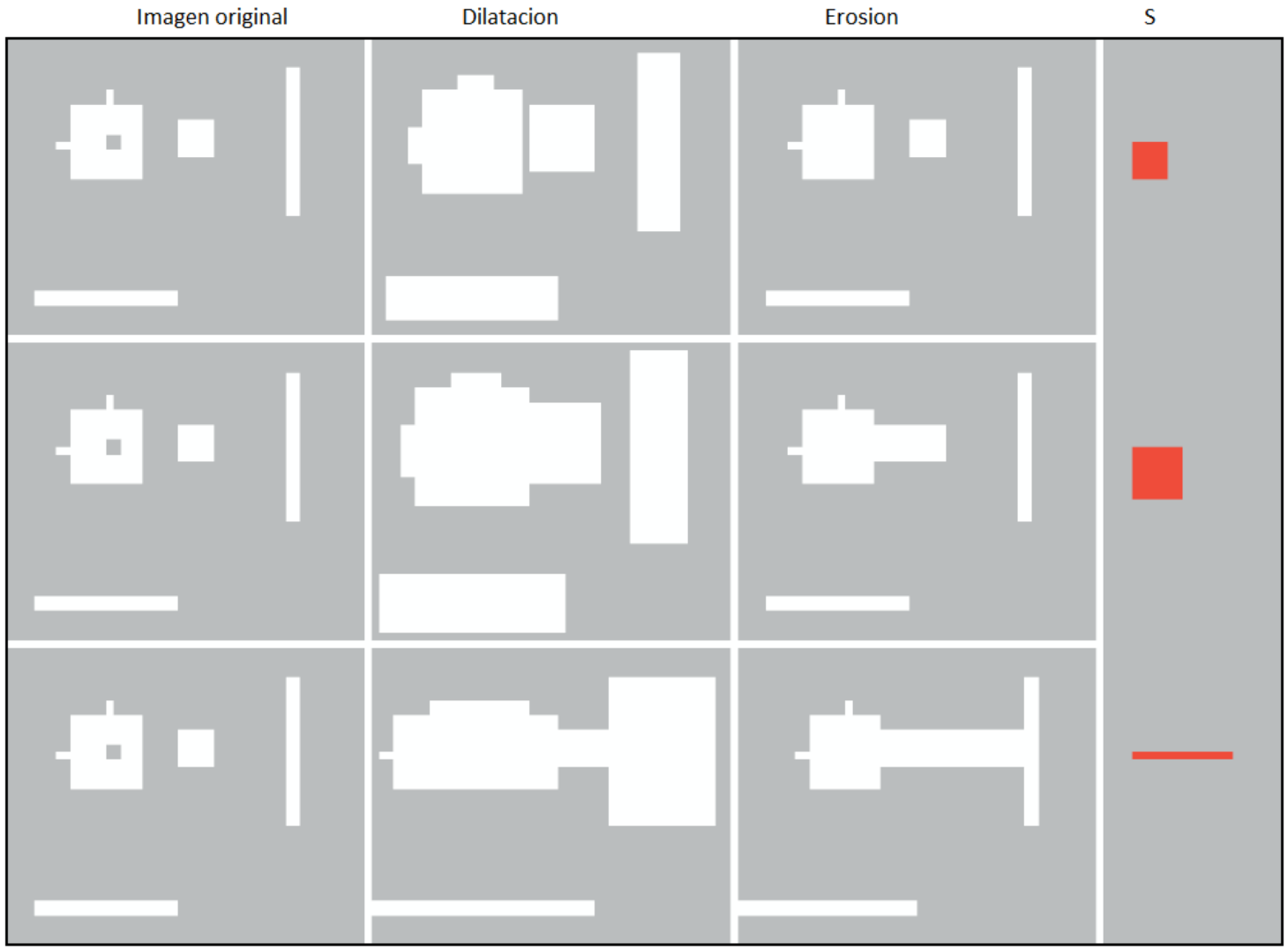


Operaciones de apertura



Iopen()

Operaciones de cierre

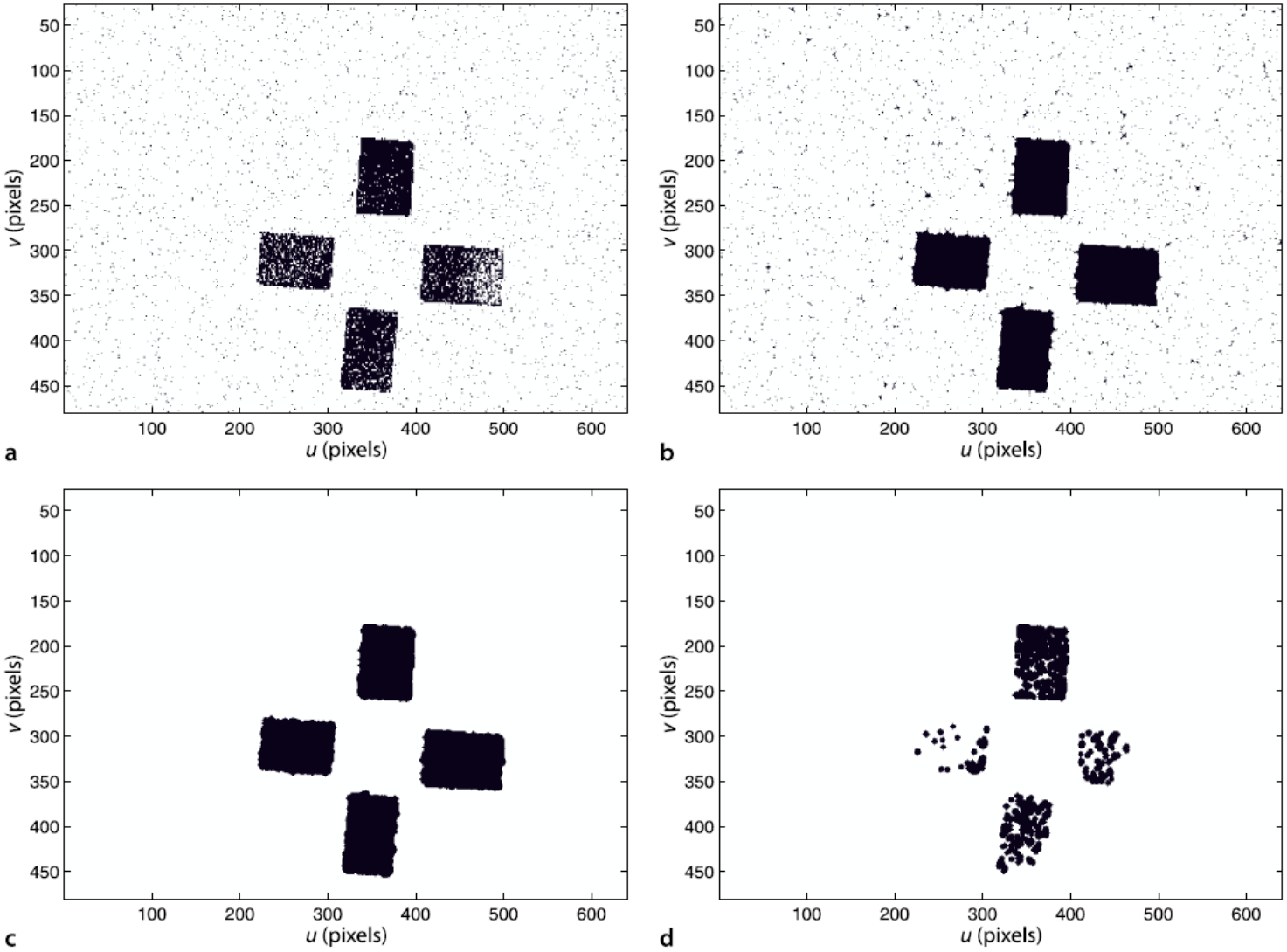


Iclose()

Aplicación: Filtrado Ruido

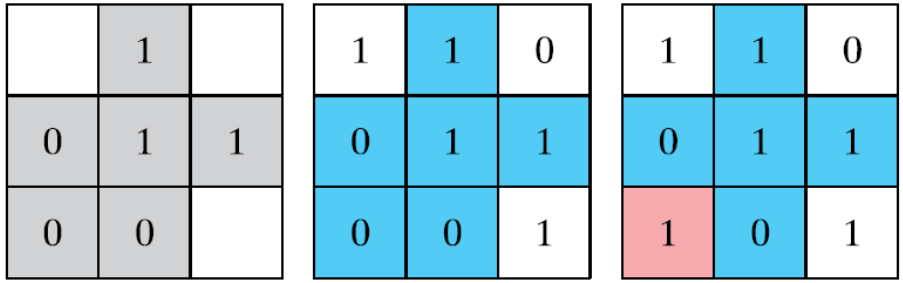
```
>> objects = iread('segmentation.png');   (a)
>> S = kcircle(3)
S =
     0     0     0     1     0     0     0
     0     1     1     1     1     1     0
     0     1     1     1     1     1     0
     1     1     1     1     1     1     1
     0     1     1     1     1     1     0
     0     1     1     1     1     1     0
     0     0     0     1     0     0     0
>> closed = iclose(objects, S);   (b)
>> clean = iopen(closed, S);   (c)
>> opened = iopen(objects, S);
>> closed = iclose(opened, S);   (d)
```

Aplicación: Filtrado Ruido

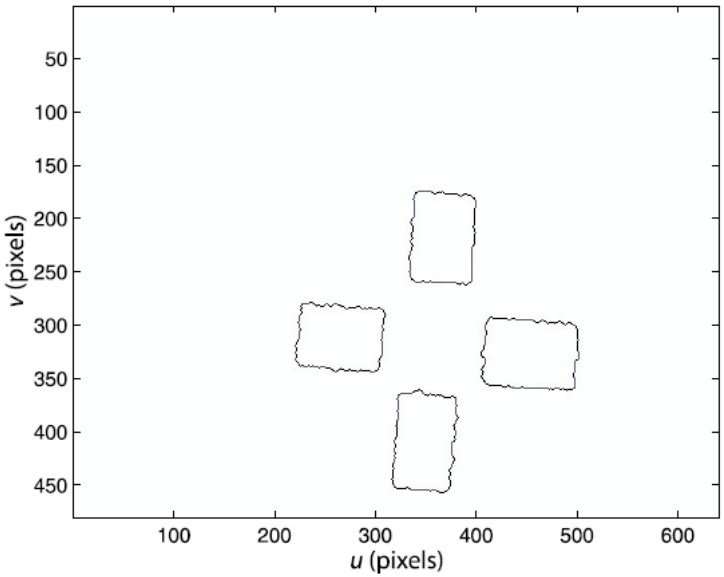


Morfología: Otras aplicaciones

- Ver 'Hit or Miss'

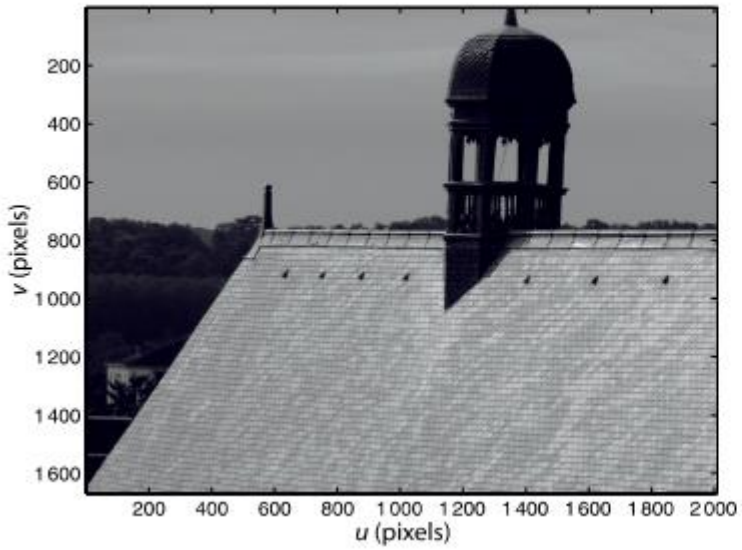


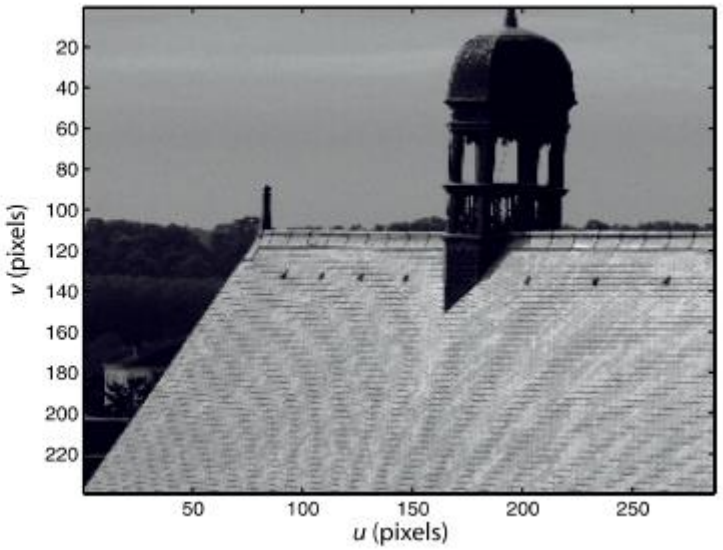
- Ver detección de bordes



Modificación de Tamaño: Sampling

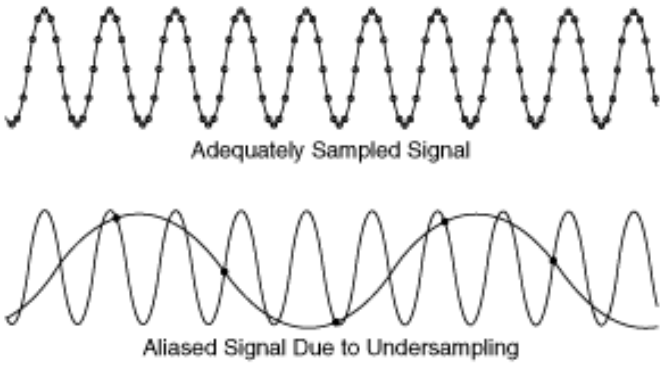
- En algunas aplicaciones es de interes reducir el tamaño de las imagenes, principalmente para disminuir su peso.
- Una técnica habitual para esto es el sampleo.
- Problemas de 'Aliasing'.



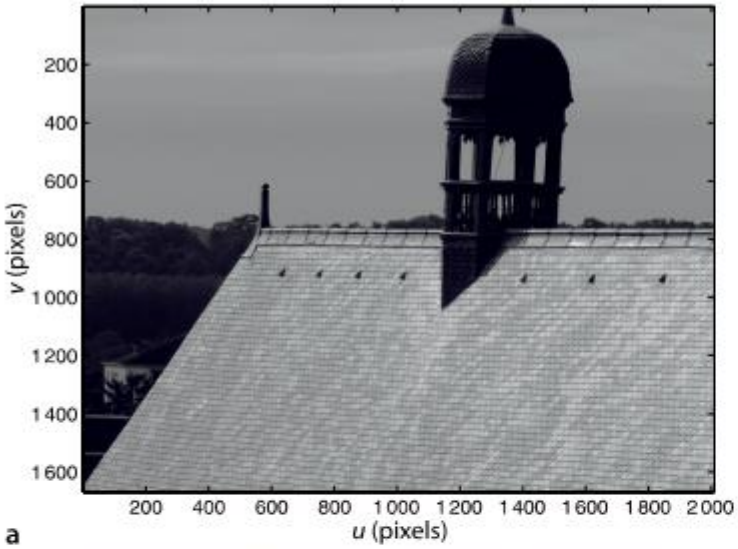


Modificación de Tamaño: Aliasing

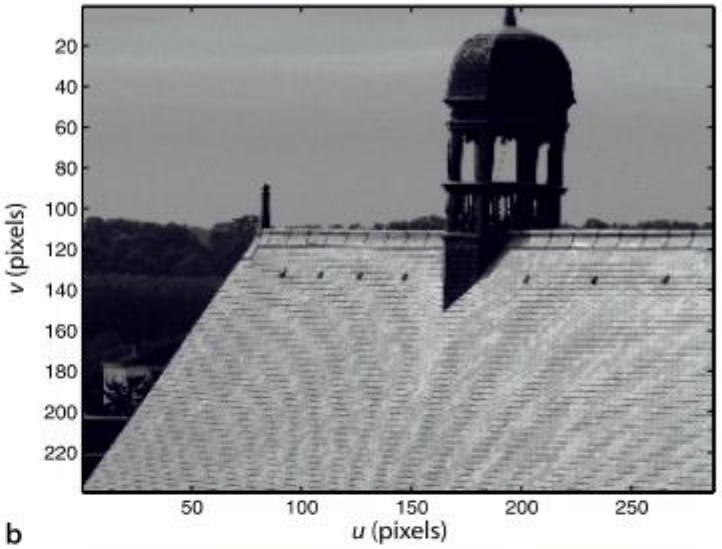
- El Aliasing es un defecto que se genera en las imágenes sampleadas. El mismo se debe al subsampleo sobre el ruido en la imagen.
- Este problema puede resolverse fácilmente, aplicando un filtro antes de realizar el sampleo.



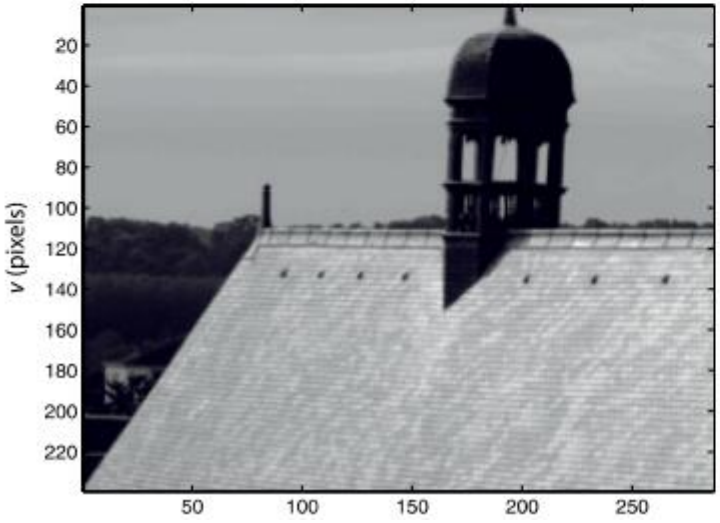
Modificación de Tamaño: Sampling

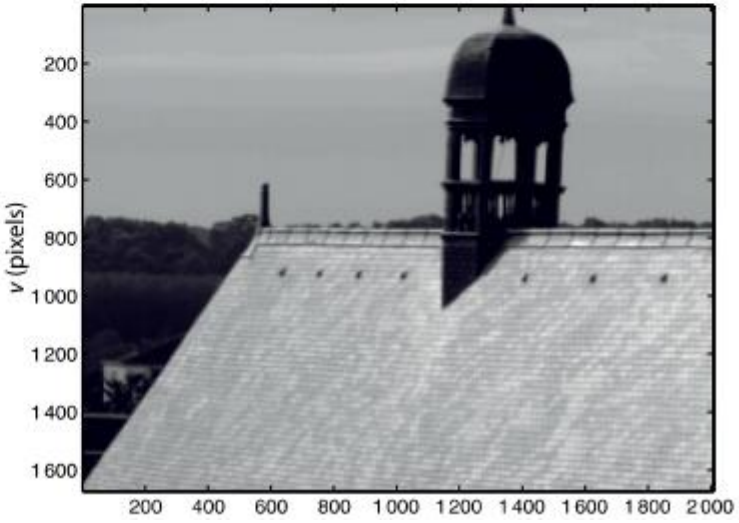


a



b

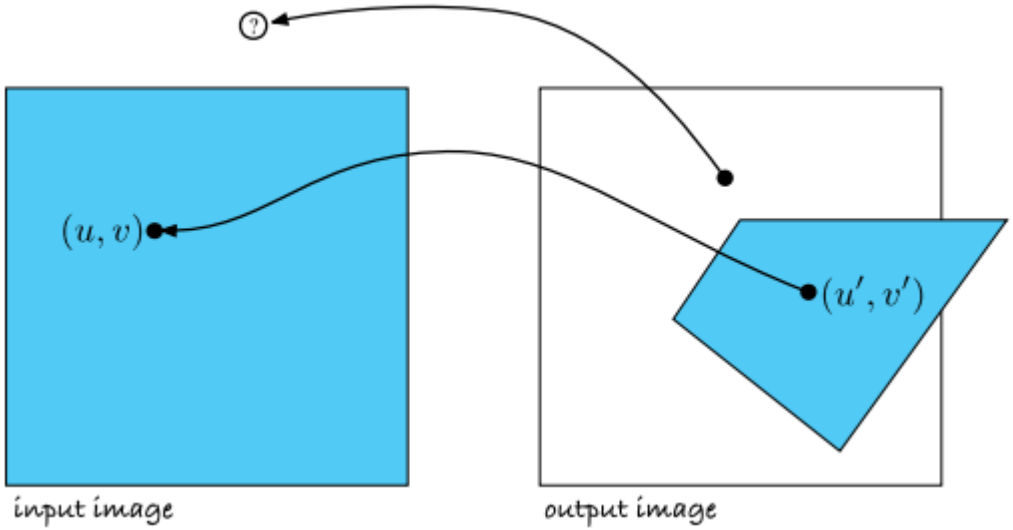





```
>> smaller = roof(1:7:end,1:7:end);
```

Modificación Generalizada: Warping

Las operaciones conocidas como 'warping' buscan modificar posición, tamaño e inclinación. Para esto, aplican funciones que en vez de modificar los valores de pixeles, modifican los valores de u y v (posición).



$$u' = f_u(u, v), \quad v' = f_v(u, v)$$

Modificación Generalizada: Warping

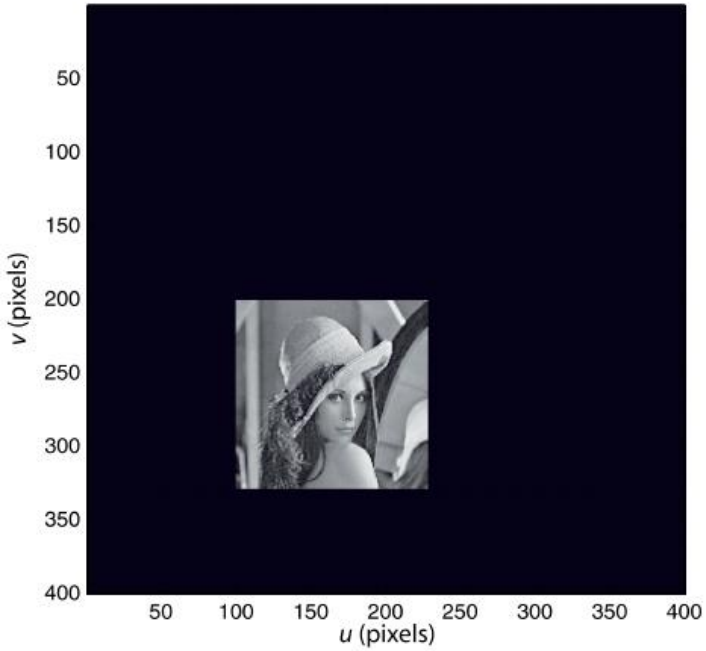
Ejemplo: cambio de tamaño y movimiento.

$$u' = u/4 + 100, \quad v' = v/4 + 200$$

```
>> [Ui,Vi] = imeshgrid(lena);
>> [Uo,Vo] = imeshgrid(400, 400);
```

$$u = 4(u' - 100), \quad v = 4(v' - 200)$$

```
>> U = 4*(Uo-100); V = 4*(Vo-200);
>> lena_small = interp2(Ui, Vi, idouble(lena), U, V);
```



Ver ejemplo Corke: Rotación

Momentos de una Figura

Hasta ahora, nos hemos focalizado en métodos orientados al procesamiento de una imagen (contraste, filtros de ruido, resaltar bordes, detectar similitudes, etc), sin embargo, es esto suficiente para una aplicación robotica?

Las imágenes son conjuntos de datos de dimensión n*m (>1000 datos), pero en un manipulador robótico la cantidad de datos de entrada que puedo admitir como en el sistema de control es sumamente inferior, incluso en manipuladores de alta complejidad.

Momentos de una Figura

Por este motivo comenzaremos a analizar métodos que se utilizan para 'discretizar' la información contenida en una imagen.

Para comenzar, estudiaremos el concepto de momentos en una imagen, el cual tiene una equivalencia directa con el calculo de momentos de una figura bidimensional.

Momentos de una Figura

- Para este análisis, trabajaremos con imágenes de tipo lógicas (aunque se pueden realizar equivalencias en imágenes no procesadas).
- Que es el momento?

$$M_{pq} = \sum u^q v^q \, I[u, v]$$

- En MATLAB

>> Mpq=mpq(im,p,q);

Aplicaciones de Momentos

- Momento de orden cero (p=q=0)

$$M_{00} = \sum u^0 v^0 \, I[u,v] = Area$$

Para p y q iguales a cero, el calculo de momento es la sumatoria de los valores de I. En el caso de una imagen de tipo lógico, esta sumatoria devuelve cuantos pixeles están 'ocupados', lo que es equivalente al área de la imagen.

Esto será de elevada importancia para el análisis de Blobs de una imagen (tema de la próxima clase)

Aplicaciones de Momentos

- Momentos de primer orden (p=1 q=0 y viceversa)

$$M_{10} = \sum u^1 v^0 \, I[u,v]$$

$$M_{01} = \sum u^0 v^1 \, I[u,v]$$

- Calculo de centroides de la imagen:

$$U_c = \frac{M_{10}}{M_{00}} \qquad V_c = \frac{M_{01}}{M_{00}}$$