Sistemas de Percepción

Práctica 1. Modelo de formación de la imagen

Poner en práctica los conocimientos adquiridos en el bloque 1 de la asignatura sobre la formación de las imágenes

Nombre: Álvaro García Lora

Curso: 2022/ 2023

Titulación: 4º Curso, GIERM

Memoria. Práctica 1.

La primera práctica de la asignatura tiene como objetivo principal poner en práctica los contenidos teóricos sobre formación de imágenes vistos en el bloque 1. Se pretende, a partir de la definición de un ejemplo plano formado por puntos, trabajar en la disposición relativa entre dicho objeto y una cámara para lograr obtener la imagen resultante de los puntos del objeto. Todo el proceso se desarrollará mediante Matlab.

Procedemos a comentar los ficheros de Matlab desarrollados.

En primer lugar, analizaremos el script ***formacionImagen.m***. Como es típico, comenzamos borrando todas las variables que componen el workspace, cerrando todas las figuras que pudieran estar abiertas y limpiando la ventana de comandos.

```
clear all; close all; clc;
```

PARÁMETROS (experimentos)

```
% Activación de efectos de distorsion
distorsion=0; % 0 == No distorsión || 1 == Distorsión

%Posición y orientación relativa de {C} respecto a {W}
dx=3;       %Distancia entre cámara y objeto respecto x_w
dy=1.5;     %Distancia entre cámara y objeto respecto y_w
dz=10;      %Distancia entre cámara y objeto respecto z_w
psi   = 0;  % Ángulo de rotación respecto a eje Z.
theta = 0;  % Ángulo de rotación respecto a eje Y'.
phi   = pi; % Ángulo de rotación respecto a eje X''.
```

En estas primeras líneas del programa podemos seleccionar si queremos tener en cuenta los efectos de distorsión que introduce la lente de la cámara o si de lo contrario queremos ignorarlos. Además, podemos modificar la posición y orientación relativa de la cámara respecto al objeto.

DEFINICIÓN DEL OBJETO (Puntos a proyectar)

En la siguiente parte del código se genera el modelo geométrico del objeto con el que trabajaremos. Como se propone, consiste en una matriz de puntos todos ellos contenidos en el plano XY del sistema de coordenadas universal {W}.

Se ha parametrizado de tal forma que en cualquier momento podemos elegir tener un objeto distinto. Mediante el cambio de nx y ny se puede asignar el tamaño del objeto rectangular y por tanto el numero de puntos totales por los cuales está formado. Además, la separación entre los puntos se puede modificar con los parámetros inc_x e inc_y.

```
nx=13;  %Número de posiciones en x
ny=7;   %Número de posiciones en y
nt=nx*ny; %Número total de puntos
inc_x=0.5; %distancia entre puntos en x (ejes W)
inc_y=0.5; %distancia entre puntos en y (ejes W)
MP=zeros(3,nx*ny); %Matriz de puntos respecto a W
```

```
x=0;y=0;k=1;
for j=1:ny
    for i=1:nx
        MP(1,k)=x;
        MP(2,k)=y;
        MP(3,k)=0;
        x=x+inc_x;
        k=k+1;
    end
    y=y+inc_y;
    x=0;
end
```

PARÁMETROS DE LA CÁMARA

```
f = 0.0042; %Distancia focal
N = 4128;   %Resolución de la imagen (ancho)
M = 3096;   %Resolución de la imagen (alto)
anchoSensor = 0.00496;  %Tamaño del sensor (ancho)
altoSensor  = 0.00352;  %Tamaño del sensor (alto)
rho_x = anchoSensor/N;  %Dimensión efectiva del píxel (horizontal)
rho_y = altoSensor/M;   %Dimensión efectiva del píxel (vertical)
fx = f/rho_x;   %Longitud focal efectiva horizontal
fy = f/rho_y;   %Longitud focal efectiva vertical
s = 0;   % Skew
u0 = round(N/2)+1;  %Punto principal horizontal de la imagen
v0 = round(M/2)-2;  %Punto principal vertical de la imagen

%Coeficientes de distorsion radial
kr1=0.144; kr2=-0.307;

%Coeficientes de distorsion tangencial
kt1=-0.0032 ; kt2=0.0017;

% Matriz de proyección simple:
Kf = [ f 0 0 0
       0 f 0 0
       0 0 1 0 ];

% Matriz de parámetros intrísecos:
K = [ fx  s*fx   u0
       0    fy   v0
       0     0    1 ];
```

En esta parte del código, además de definir los parámetros de la cámara a usar, se realiza el cálculo de magnitudes características que posteriormente serán utilizadas en el proceso de proyección de los puntos del objeto a la imagen. En este sentido, también se crean las matrices de proyección simple (Kf) y de parámetros intrínsecos (K).

POSICIÓN RELATIVA ENTRE CÁMARA Y OBJETO

A partir de los parámetros definidos en el principio del programa (dx.dy,dz,psi,theta y phi) que definían la posición y orientación del SR asociado la cámara {C} respecto al universal {W} realizamos el cálculo de la matriz de transformación homogénea que lleva de {W} a {C} como se muestra:

```
% Posición del origen de {C} respecto al sistema {W}:
wtc = [dx,dy,dz]';

% Orientación {C} respecto al sistema {W}. En general dada por sucesión de tres giros
(convenio ZYX ejes móviles):

Rz = [ cos(psi) -sin(psi) 0
       sin(psi)  cos(psi) 0
           0         0    1 ];

Ry = [ cos(theta) 0 sin(theta)
           0      1    0
      -sin(theta) 0 cos(theta) ];

Rx = [ 1    0        0
       0 cos(phi) -sin(phi)
       0 sin(phi)  cos(phi) ];

wRc = Rz*Ry*Rx; % Matriz de rotación {C} a {W}

wTc = [wRc wtc; [0 0 0 1]]; % Matriz de transformación homogénea {C} a {W}

% Obtenemos cTm a partir de wTc obteniendo su inversa
%cTw = inv(wTc); % Matriz de transformación homogénea {W} a {C}
cTw = [wRc', -wRc'*wtc; [0 0 0 1]]; % Forma más eficiente
cRw = cTw(1:3,1:3); % Matriz de rotación de la transformación {W} a {C}
ctw = cTw(1:3,4);   % Vector de traslación de la transformación {W} a {C}
```

Proyección de los puntos del objeto a la imagen

```
%CON DISTORSIÓN
if(distorsion==1)
    MP_ = [MP ; ones(1,nt)]; %Matriz de puntos en homogeneas
    mpc_ = Kf * cTw * MP_ ;  %Matriz de puntos en coordenadas homogeneas respecto a {C}
    mpc(1:2,:)= mpc_(1:2,:)./mpc_(3,:); %Matriz puntos en coordenadas estandar respecto a {C}
    mpDist = zeros(size(mpc));
    for i=1:nt
        pNorm = mpc(:,i)/f;  %Pasa el punto a coordenadas proyectadas normalizadas
        r = norm(pNorm);     %Cálculo del radio
        deltaR = 1 + kr1*r^2 + kr2*r^4; % Distorsion radial
        deltaTx=2*kt1*pNorm(1)*pNorm(2)+kt2*(r^2+2*pNorm(1)^2); % Distorsion tangencial en x
        deltaTy=2*kt2*pNorm(1)*pNorm(2)+kt1*(r^2+2*pNorm(2)^2); % Distorsion tangencial en y
        mpDist(:,i) = ((pNorm * deltaR) + [deltaTx ; deltaTy])*f; %Matriz puntos distorsion
    end
    mp(1,:)= round(mpDist(1,:)/rho_x + ones(1,nt)*u0); % Discretizacion puntos de la imagen
    mp(2,:)= round(mpDist(2,:)/rho_y + ones(1,nt)*v0); % Discretizacion puntos de la imagen
```

```
%SIN DISTORSIÓN
elseif(distorsion==0)
    MP_ = [MP ; ones(1,nt)]; %Matriz de puntos en homogeneas
    mp_ = K * [cRw ctw] * MP_ ; %Matriz puntos coordenadas homogeneas respecto {C} (pixeles)
    mp(1:2,:)= round(mp_(1:2,:)./mp_(3,:)); %Matriz puntos coordenadas estandar respecto {C}
(en pixeles)
end
```

En función del valor del parámetro distorsión vemos como se tienen en cuenta o no los efectos de distorsión de la lente. Los pasos seguidos para realizar la proyección a puntos en la imagen son los estudiados en el bloque 1, los cuales se encuentran comentados con detalle en el propio código mostrado como podemos observar.

REPRESENTACIONES

```
figura3d(MP(1,:),MP(2,:),MP(3,:),wTc) %Objeto y Cámara en el espacio
figuraImagen(mp(1,:),mp(2,:),M,N,nx,ny) %Imagen generada en la cámara
```

Para ilustrar de manera gráfica el trabajo realizado se han desarrollado dos funciones encargadas de esta tarea. Dichas funciones deben estar incluidas en el mismo directorio de trabajo que el script principal que hemos estado comentando.

La primera de ellas, ***figura3d.m*** representa el objeto y la cámara en el espacio. Gracias a ella podemos observar con claridad la posición relativa entre ambos.

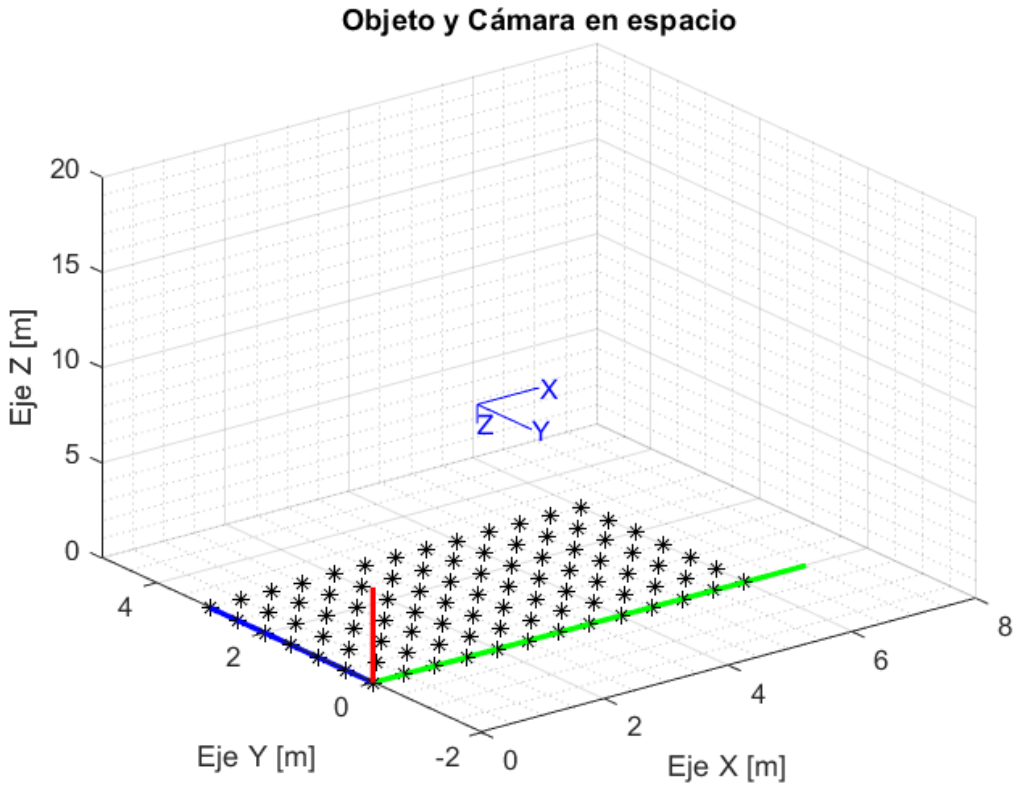
```
function figura3d(X, Y, Z, T)

%  X:  vector datos x
%  Y:  vector datos y
%  Z:  vector datos z
%  T: MTH de C respecto W

% Creamos figura
figure1 = figure;

% Creamos ejes
axes1 = axes('Parent',figure1);
hold(axes1,'on');

% Creamos plot3
trplot(T); %Representa ejes de la cámara {C}
plot3(X,Y,Z,'k','Marker','*','LineStyle','none'); %Objeto
plot3([0,7],[0,0],[0,0],'g','LineWidth',2); %eje x de {W}
plot3([0,0],[0,3],[0,0],'b','LineWidth',2); %eje y de {W}
plot3([0,0],[0,0],[0,5],'r','LineWidth',2); %eje z

% zlabel
zlabel({'Eje Z [m]'});

% ylabel
ylabel({'Eje Y [m]'});

% xlabel
xlabel({'Eje X [m]'});
```

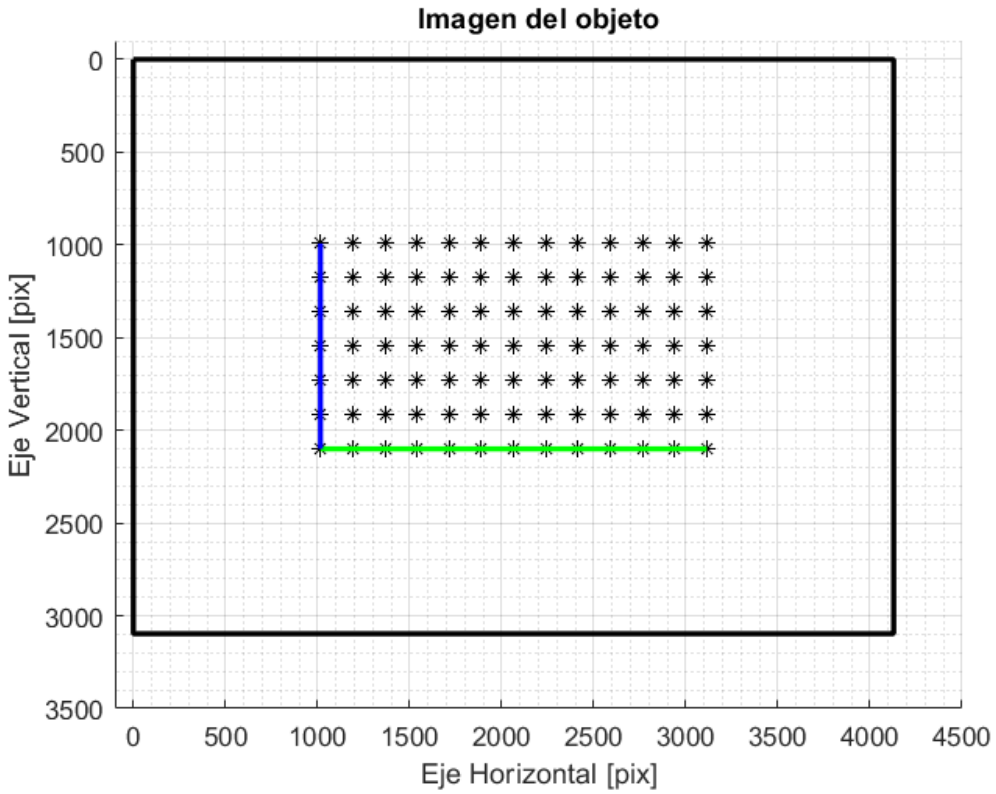
```
% Título
title({'Objeto y Cámara en espacio'});

xlim(axes1,[0 8]);
ylim(axes1,[-2 5]);
zlim(axes1,[0 20]);
view(axes1,[-37.5 30]);

grid(axes1,'on');
hold(axes1,'off');

set(axes1,'XMinorGrid','on','YMinorGrid','on','ZMinorGrid','on');
```

Por otro lado, ***figuraImagen.m*** muestra la imagen que se genera en la cámara, es decir, en otras palabras, representa la proyección de los puntos del objeto en el plano de la imagen de la cámara.

```
function figuraImagen(X, Y, M,N,nx,ny)

%  X:  vector datos x
%  Y:  vector datos y

% Creamos figura
figure1 = figure;

% Creamos ejes
axes1 = axes('Parent',figure1);
hold(axes1,'on');

% Creamos plot
plot(X,Y,'k','Marker','*','LineStyle','none');
plot([0,0,N+1,N+1,0],[0,M+1,M+1,0,0],'Color','k', 'LineWidth',2);
plot([X(1),X(nx)],[Y(1),Y(nx)],'g','LineWidth',2); %eje x_w
plot([X(1),X((nx*(ny-1))+1)],[Y(1),Y((nx*(ny-1))+1)],'b','LineWidth',2); %eje y_w

% ylabel
ylabel({'Eje Vertical [pix]'});

% xlabel
xlabel({'Eje Horizontal [pix]'});

%  Título
title({'Imagen del objeto'});

xlim(axes1,[-100 4500]);
ylim(axes1,[-100 3500]);

grid(axes1,'on');
hold(axes1,'off');

set(axes1,'XMinorGrid','on','YMinorGrid','on','ZMinorGrid','on');
set(gca,'YDir', 'reverse');
```

Representamos como ejemplo la disposición de la cámara perfectamente enfrentada al objeto (parámetros que se muestran en el código comentado anteriormente, sin distorsión y con distancia al mismo de 10 m):



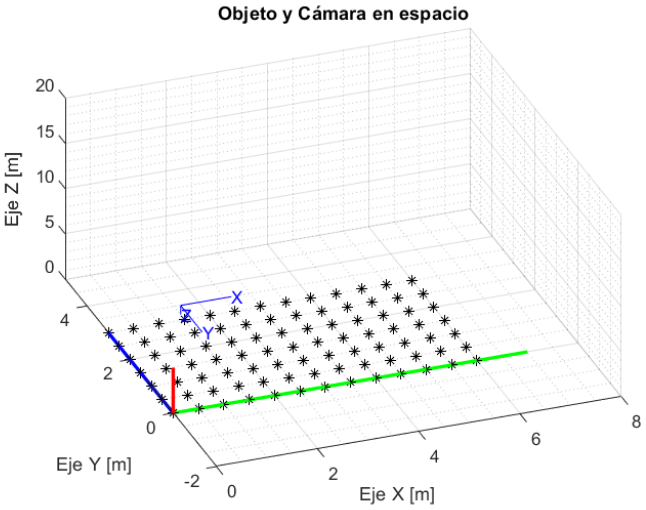


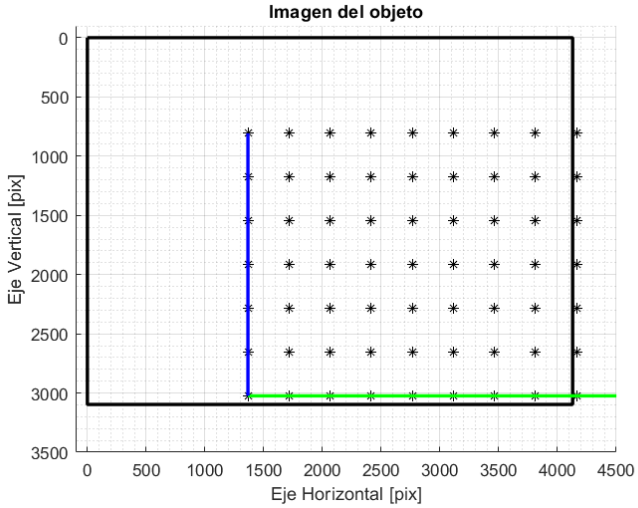
PRUEBAS REALIZADAS

A continuación, se muestra un conjunto de pruebas realizadas donde se observan los resultados de la proyección del objeto en la imagen al variar aspectos básicos como son la posición y orientación relativa objeto-cámara, la distancia focal o los parámetros de distorsión.

- Reducimos la distancia entre cámara y objeto respecto al caso inicial visto y variamos a la vez las coordenadas x, y del punto origen de {C}:

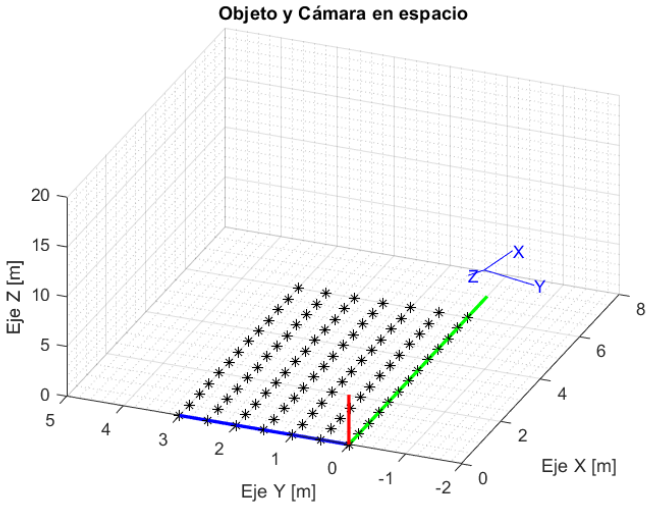
```
%Posición y orientación relativa de {C} respecto a {W}
dx=1;       %Distancia entre cámara y objeto respecto x_w
dy=2;     %Distancia entre cámara y objeto respecto y_w
dz=5;      %Distancia entre cámara y objeto respecto z_w
```

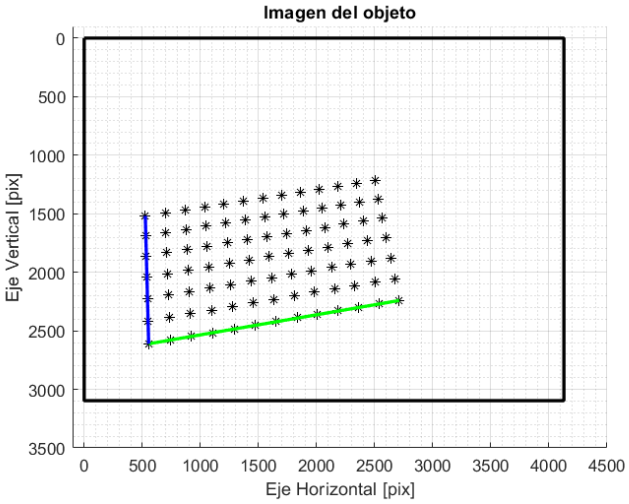




- Cambiamos esta vez la posición y orientación, además incluimos el efecto de distorsión:

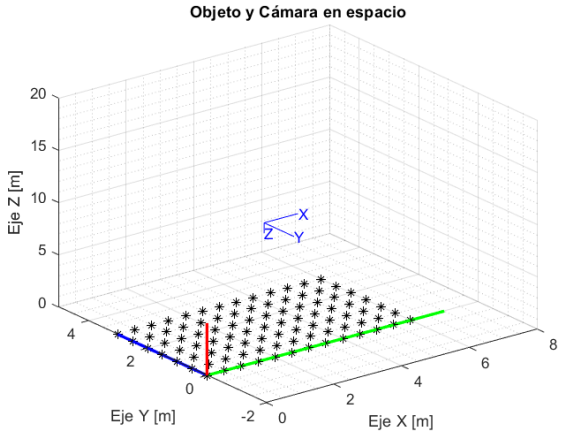
```
% Activación de efectos de distorsion
distorsion=1; % 0 == No distorsión || 1 == Distorsión

%Posición y orientación relativa de {C} respecto a {W}
dx=4;       %Distancia entre cámara y objeto respecto x_w
dy=-1;     %Distancia entre cámara y objeto respecto y_w
dz=10;      %Distancia entre cámara y objeto respecto z_w
psi   = -pi/20;  % Ángulo de rotación respecto a eje Z.
theta = 0;  % Ángulo de rotación respecto a eje Y'.
phi   = pi + pi/10; % Ángulo de rotación respecto a eje X''.
```

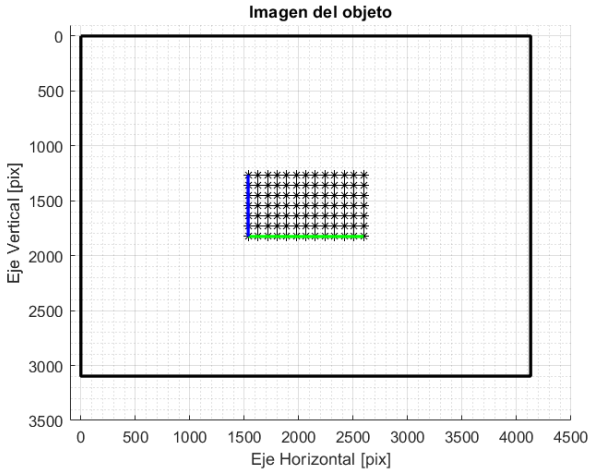




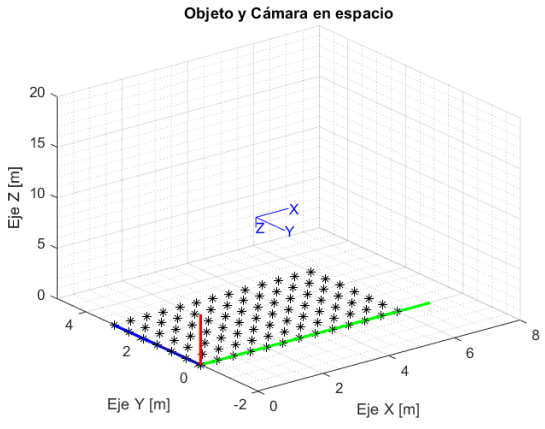
- Variamos la distancia focal a la mitad, con distorsión y en la posición inicial:

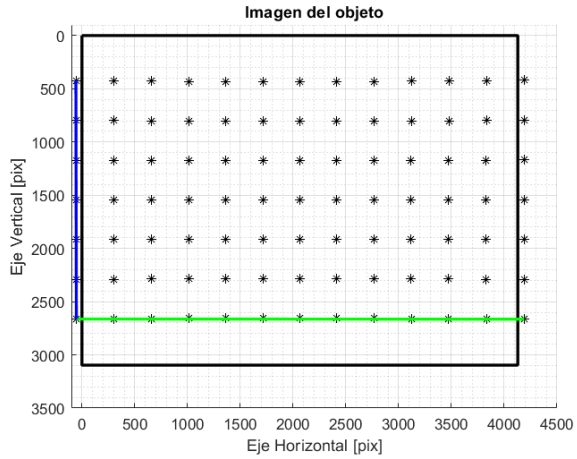
```
f = 0.0042 * 0.5; % Distancia focal
```






```
f = 0.0042 * 2; % Distancia focal
```

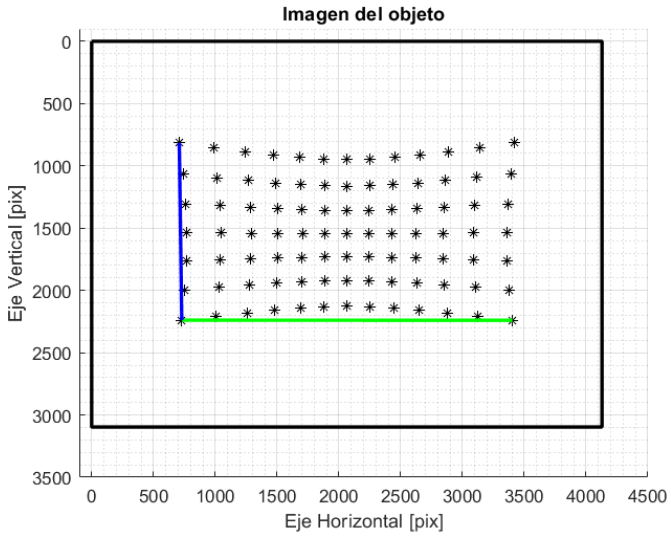




Podemos ver como la variación de la distancia focal supone un efecto de zoom en la imagen final, es equivalente a variar la distancia en el eje óptico, es decir, la separación entre la cámara y el objeto.

- Por último, alteramos los parámetros de distorsión (la posición y orientación corresponde a la inicial):
    - Aumentamos kr1 en 20 veces el valor inicial y kr2 en 10 veces.

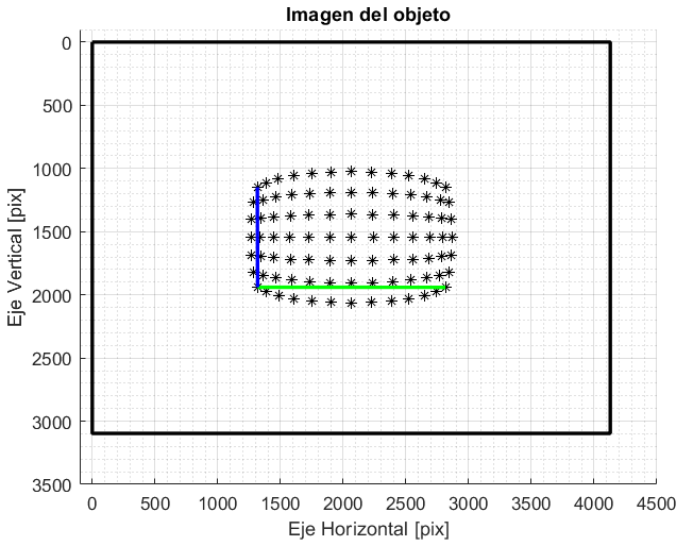
```
%Coeficientes de distorsion radial
kr1=0.144*20; kr2=-0.307*10;

%Coeficientes de distorsion tangencial
kt1=-0.0032; kt2=0.0017;
```



Distorsión de tipo “pincushion” o cóncava

- Cambiamos el signo de los parámetros anteriores:

```
%Coeficientes de distorsion radial
kr1=-0.144*20; kr2=0.307*10;

%Coeficientes de distorsion tangencial
kt1=-0.0032; kt2=0.0017;
```



Distorsión de tipo “barrel” o biconvexa