UNCUYO
UNIVERSIDAD NACIONAL DE CUYO

FACULTAD DE INGENIERÍA

Trabajo Final
Inteligencia Artificial I

Alumno: Víctor Silva

Carrera: Ingeniería en Mecatrónica

Legajo: 10988

Año: 2019

Trabajo Final – Inteligencia Artificial I - 2019

Resumen

El trabajo consiste en un sistema de clasificación de piezas metálicas por visión artificial. Para ello se propone desarrollar una agente que permita identificar tornillos, clavos, tuercas y arandelas. Para llevar a cabo esta tarea se recurre a la utilización de dos algoritmos empleados en machine learning, uno de ellos es KNN (K nearest neighbors) y el otro es K-means. Se analizará el tipo de agente, su tabla REAS y su entorno de trabajo. Luego se analizarán y compararán los resultados obtenidos del problema propuesto.

Introducción

Para este problema se ha implementado un ***agente que aprende***. Además de las percepciones de su entorno y tener la capacidad de actuar como otros agentes, tiene la cualidad de que puede aprender de sus experiencias para actuar de forma más eficiente en próximas experiencias.

La tabla REAS del agente es la siguiente:

Agente	Medida de rendimiento	Entorno	Actuadores	Sensores
Clasificador de piezas con visión artificial	Cantidad de piezas separadas correctamente Velocidad de procesamiento de la imagen	Cinta transportadora	Motores para mover la pieza a la posición correspondiente	Cámara Posición

Las propiedades del entorno de trabajo son:

Entorno de trabajo	Observable	Determinista	Episódico	Estático	Discreto	agente
Robot clasificando en cinta transportadora	Parcialmente observable	No. Estocástico	Sí	No. Dinámico	No. Continuo	Individual

Diseño del Sistema

En primer lugar, se selecciona diez imágenes de cada uno de los distintos tipos de piezas a analizar para formar la base de conocimiento de las imágenes de las piezas metálicas, las cuales siguen el siguiente proceso: adquisición de la imagen, preprocesamiento, segmentación, extracción de características. Los resultados de los parámetros de estas se guardaron un archivo de texto para disminuir el tiempo de cómputo.

Luego se corre el programa main, en el cual se ingresa el nombre de la imagen a clasificar. Una vez ingresado se muestra por pantalla y se consulta si es adecuada la visualización de ella, en caso de que lo sea, seleccione “S” y se continua con el programa, en caso contrario se pueden modificar algunos parámetros como el nivel de im2bw y seleccionar si esta imagen tiene un fondo blanco o no, para que quede el fondo de color negro. Una vez aceptada la imagen se le extraen las características, las cuales son: área de la pieza, valor máximo de la suma de cada uno de los elementos a lo largo de una fila y columna anterior al número del elemento de la diagonal y la comparación entre el área de pieza multiplicado por un factor y un cuadrado que va aumentado en cada iteración hasta alcanzar el valor del producto antes mencionado del área.

Posteriormente se le aplican los algoritmos KNN y K-menas. A KNN se le puede seleccionar el valor de k para conocer esa cantidad de vecinos más cercanos. En caso de que la cantidad de más cercanos de una clase sea igual a otra se debe volver a seleccionar el valor de k. En el caso de k-means se dejó fijo el valor de k, ya que es necesario tener las 4 clases establecidas. Finalmente me dice que es lo que se reconoció para cada algoritmo y se muestran de forma gráfica el proceso cuando se ingresa e imagen y cuando ya es clasificada.

Código

```
clc,clear

disp('Se aplicaran los algoritmos K-means y KNN a la imagen introducida para
clasificarla');
imagen=input('Ingrese el nombre del la imagen a analizar sin el formato:
','s');
nivel=0.55;
negro=1;
flag=0;
parametros=0;
while flag==0
    [nuev]= nueva_imagen(imagen,nivel,negro,parametros);
    disp('La imagen debe tener fondo de color negro')
    disp('¿Resulta satisfactoria la imagen segmentada?');
    correcta=input('(S)Es correcta /(N)No es correcta: ','s');
    if (correcta=='S')||(correcta=='s')
        flag=1;
    else
        flag2=0;
        while flag2==0
            fprintf('El valor actual de la nivel es de %f \n',nivel);
            nivel=input('Ingrese un valor para modificar la nivel comprendido
entre 0 y 1: ');
            if (nivel<0 || nivel>1)
                disp('Error, debe estar compendido en el intervalo');
            else
                flag2=1;
            end
            disp('¿El fondo de la imagén está todo blanco?');
            demasiado_blanca=input('(S)Sí /(N)No: ','s');
            if (demasiado_blanca=='S')||(demasiado_blanca=='s')
                negro=0;
            end
        end
    end
end
parametros=1;
[nuev]= nueva_imagen(imagen,nivel,negro,parametros);
disp('A continuación se aplicaran los algoritmos K-means y KNN a la imagen
introducida para clasificarla');
fprintf('\n');
%----------KKN----------
disp('Algoritmo KNN');
flag3=0;
while(flag3==0)
    k=input('Ingrese el valor de k del algoritmo KNN: ');
    error=Knn(k,nuev);
    if error==0
        flag3=1;
    end
end
%----------K-MEANS----------
disp('Algoritmo K-means');
Kmeans(nuev);
```

```
function [nuev]= nueva_imagen(imagen,nivel,negro,parametros)

n = genvarname({'imagen'});
Nueva = genvarname({char(imagen)});
n1 = genvarname({'nuevai'});
n2 = genvarname({'nuevabin'});

texto=strcat(Nueva(1),'.jpg');
n{1} =imread(char(texto));
n1{1}=imresize(n{1},[500 500]);
n2{1}=im2bw(n1{1},nivel);
if negro==0
    n2{1} = 1 - n2{1};
end

if parametros==0
    imshow(n1{1})
    figure
    imshow(n2{1})
end

nuev=zeros(3,1);
if parametros==1
    vectn=genvarname({'vectn'});
    vectn{1}=zeros(1,length(n2{1}));

    for i=1:500
        for j=1:500
            if i>=j
                vectn{1}(i)=vectn{1}(i)+n2{1}(i,j);
            else
                vectn{1}(i)=vectn{1}(i)+n2{1}(i,j);
            end
        end
    end
    %------------------PARAMETRO 1--------------------

    nuev(1,1)= max(vectn{1});

    %------------------PARAMETRO 2--------------------
    for i=1:1:500
        for j=1:1:500
            nuev(2,1)= n2{1}(i,j)+nuev(2,1);
        end
    end

    %-----------------PARAMETRO 3:--------------------
    factor=0.6;
    for j=1:249
        aux=0;
        for m=-j:j
            for n=-j:j
                if n2{1}(250+m,250+n)==1
                    aux=aux+1;
                end
```

```
            end
        end
        if aux>nuev(2,1)*factor
            nuev(3,1)= j;
            break
        end
    end
    nuev(2,1)=nuev(2,1)/1000;
end
```

```
function [vect,color]=entrenar_base()

a = genvarname({'aran', 'aran', 'aran', 'aran', 'aran', 'aran', 'aran',
'aran', 'aran', 'aran'});
Arandela = genvarname({'Arandela', 'Arandela', 'Arandela', 'Arandela',
'Arandela', 'Arandela', 'Arandela', 'Arandela', 'Arandela', 'Arandela'});
a1 = genvarname({'arani', 'arani', 'arani', 'arani', 'arani', 'arani',
'arani', 'arani', 'arani', 'arani'});
a2 = genvarname({'aranbin', 'aranbin', 'aranbin', 'aranbin', 'aranbin',
'aranbin', 'aranbin', 'aranbin', 'aranbin', 'aranbin'});
l1=length(a)
for i=1:l1
    text=strcat(Arandela(i),'.jpg');
    a{i} =imread(char(text));
    a1{i}=imresize(a{i},[500 500]);
    a2{i}=im2bw(a1{i},0.55);
    %figure
    %imshow(a2{i})
end

to = genvarname({'torn', 'torn', 'torn', 'torn', 'torn', 'torn', 'torn',
'torn', 'torn', 'torn'});
Tornillo = genvarname({'Tornillo', 'Tornillo', 'Tornillo', 'Tornillo',
'Tornillo', 'Tornillo', 'Tornillo', 'Tornillo', 'Tornillo', 'Tornillo'});
to1 = genvarname({'torni', 'torni', 'torni', 'torni', 'torni', 'torni',
'torni', 'torni', 'torni', 'torni'});
to2 = genvarname({'tornibin', 'tornibin', 'tornibin', 'tornibin', 'tornibin',
'tornibin', 'tornibin', 'tornibin', 'tornibin', 'tornibin'});
l2=length(to);
for i=1:l2
    text=strcat(Tornillo(i),'.jpg');
    to{i} =imread(char(text));
    to1{i}=imresize(to{i},[500 500]);
    to2{i}=im2bw(to1{i},0.55);
    %figure
    %imshow(to2{i})
end

tu = genvarname({'tuer', 'tuer', 'tuer', 'tuer', 'tuer', 'tuer', 'tuer',
'tuer', 'tuer', 'tuer'});
Tuerca = genvarname({'Tuerca', 'Tuerca', 'Tuerca', 'Tuerca', 'Tuerca',
'Tuerca', 'Tuerca', 'Tuerca', 'Tuerca', 'Tuerca'});
tu1 = genvarname({'tuerni', 'tuerni', 'tuerni', 'tuerni', 'tuerni', 'tuerni',
'tuerni', 'tuerni', 'tuerni', 'tuerni'});
tu2 = genvarname({'tuerbin', 'tuerbin', 'tuerbin', 'tuerbin', 'tuerbin',
'tuerbin', 'tuerbin', 'tuerbin', 'tuerbin', 'tuerbin'});
l3=length(tu);
for i=1:l3
    text=strcat(Tuerca(i),'.jpg');
    tu{i} =imread(char(text));
    tu1{i}=imresize(tu{i},[500 500]);
    tu2{i}=im2bw(tu1{i},0.55);
    %figure
    %imshow(tu2{i})
end
```

```
c = genvarname({'clav', 'clav', 'clav', 'clav', 'clav', 'clav', 'clav',
'clav', 'clav', 'clav'});    %3 clavos
Clavo = genvarname({'Clavo', 'Clavo', 'Clavo', 'Clavo', 'Clavo', 'Clavo',
'Clavo', 'Clavo', 'Clavo', 'Clavo'});
c1 = genvarname({'clavi', 'clavi', 'clavi', 'clavi', 'clavi', 'clavi',
'clavi', 'clavi', 'clavi', 'clavi'});
c2 = genvarname({'clavbin', 'clavbin', 'clavbin', 'clavbin', 'clavbin',
'clavbin', 'clavbin', 'clavbin', 'clavbin', 'clavbin'});
l4=length(c);
for i=1:l4
    text=strcat(Clavo(i),'.jpg');
    c{i} =imread(char(text));
    c1{i}=imresize(c{i},[500 500]);
    c2{i}=im2bw(c1{i},0.60);
    %figure
    %imshow(c2{i})
end

vecta=genvarname({'vecta','vecta','vecta','vecta','vecta','vecta','vecta','ve
cta','vecta','vecta'});
for i=1:l1
    vecta{i}=zeros(1,length(a2{i}));
end
vectto=genvarname({'vectto','vectto','vectto','vectto','vectto','vectto','vec
tto','vectto','vectto','vectto'});
for i=1:l2
    vectto{i}=zeros(1,length(to2{i}));
end
vecttu=genvarname({'vecttu','vecttu','vecttu','vecttu','vecttu','vecttu','vec
ttu','vecttu','vecttu','vecttu'});
for i=1:l3
    vecttu{i}=zeros(1,length(tu2{i}));
end
vectc=genvarname({'vectc','vectc','vectc','vectc','vectc','vectc','vectc','ve
ctc','vectc','vectc'});
for i=1:l4
    vectc{i}=zeros(1,length(c2{i}));
end

%----------PARAMETRO 1: Sumo toda las filas anteriores y columnas anteriores
para reducirlo a un elemento---------
for i=1:500
    for j=1:500
        if i>=j
            for k=1:l1
                vecta{k}(i)=vecta{k}(i)+a2{k}(i,j);
            end
            for k=1:l2
                vectto{k}(i)=vectto{k}(i)+to2{k}(i,j);
            end
            for k=1:l3
                vecttu{k}(i)=vecttu{k}(i)+tu2{k}(i,j);
            end
            for k=1:l4
                vectc{k}(i)=vectc{k}(i)+c2{k}(i,j);
            end
```

```
        else
            for k=1:l1
                vecta{k}(i)=vecta{k}(i)+a2{k}(i,j);
            end
            for k=1:l2
                vectto{k}(i)=vectto{k}(i)+to2{k}(i,j);
            end
            for k=1:l3
                vecttu{k}(i)=vecttu{k}(i)+tu2{k}(i,j);
            end
            for k=1:l4
                vectc{k}(i)=vectc{k}(i)+c2{k}(i,j);
            end

        end
    end
end

vect=zeros(3,l1+l2+l3+l4);
for i=1:(l1+l2+l3+l4)
    if (i <= l1)
        vect(1,i)= max(vecta{i});
    end
    if (i <= l1+l2 && i>l1)
        vect(1,i)= max(vectto{i-l1});
    end
    if (i <= l1+l2+l3 && i>l1+l2)
        vect(1,i)= max(vecttu{i-l1-l2});
    end
    if (i <= l1+l2+l3+l4 && i>l1+l2+l3)
        vect(1,i)= max(vectc{i-l1-l2-l3});
    end
end

%------------------PARAMETRO 2: Suma de todo el área--------------------
for i=1:1:500
    for j=1:1:500
        for k=1:(l1+l2+l3+l4)
            if k <= l1
                vect(2,k)= a2{k}(i,j)+vect(2,k);
            end
            if k <= (l1+l2) && k > l1
                vect(2,k)= to2{k-l1}(i,j)+vect(2,k);
            end
            if k <= (l1+l2+l3) && k > (l1+l2)
                vect(2,k)= tu2{k-l1-l2}(i,j)+vect(2,k);
            end
            if k <= (l1+l2+l3+l4) && k > (l1+l2+l3)
                vect(2,k)= c2{k-l1-l2-l3}(i,j)+vect(2,k);
            end
        end
    end
end

%--------------------PARAMETRO 3: Comparación con un cuadrado que aumenta de
lado-----------------------------------
```

```
factor=0.6;
for k=1:(l1+l2+l3+l4)
    if k <= l1
        for j=1:249
            aux=0;
            for m=-j:j
                for n=-j:j
                    if a2{k}(250+m,250+n)==1
                        aux=aux+1;
                    end
                end
            end
            if aux>vect(2,k)*factor
                vect(3,k)= j;
                break
            end
        end
    end
    if k <= (l1+l2) && k > l1
        for j=1:249
            aux=0;
            for m=-j:j
                for n=-j:j
                    if to2{k-l1}(250+m,250+n)==1
                        aux=aux+1;
                    end
                end
            end
            if aux>vect(2,k)*factor
                vect(3,k)= j;
                break
            end
        end
    end
    if k <= (l1+l2+l3) && k > (l1+l2)
        for j=1:249
            aux=0;
            for m=-j:j
                for n=-j:j
                    if tu2{k-l1-l2}(250+m,250+n)==1
                        aux=aux+1;
                    end
                end
            end
            if aux>vect(2,k)*factor
                vect(3,k)= j;
                break
            end
        end
    end
    if k <= (l1+l2+l3+l4) && k > (l1+l2+l3)
        for j=1:249
            aux=0;
            for m=-j:j
                for n=-j:j
                    if c2{k-l1-l2-l3}(250+m,250+n)==1
                        aux=aux+1;
```

```
                    end
                end
            end
            if aux>vect(2,k)*factor
                vect(3,k)= j;
                break
            end
        end
    end
end
vect(2,:)=vect(2,:)/1000;

color=[0 1 0;0 1 0;0 1 0;0 1 0;0 1 0;0 1 0;0 1 0;0 1 0;0 1 0;0 1 0;0 0 0;0 0
0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;1 0 0;1 0 0;1 0 0;1 0 0;1 0
0;1 0 0;1 0 0;1 0 0;1 0 0;1 0 0;1 0 0;0 1 1;0 1 1;0 1 1;0 1 1;0 1 1;0 1 1;0 1
1;0 1 1;0 1 1];

figure
hold on
scatter3(vect(1,:),vect(2,:),vect(3,:),25,color)

xlabel('valor maximo diagonal');
ylabel('área');
zlabel('cuadrado');
title('Base de conocimiento')
grid on
hold off

fileID = fopen('C:\Users\Víctor\Documents\MATLAB\IA\vect.txt','w');
fprintf(fileID,' %8.4f %8.4f %8.4f \n',vect);
fclose(fileID);
fileID = fopen('C:\Users\Víctor\Documents\MATLAB\IA\color.txt','w');
fprintf(fileID,' %8.4f %8.4f %8.4f \n',color);
fclose(fileID);

end
```

```
function [error]= Knn(k,nuev)

%----------Para entrenar otra base de conocimiento----------
%[vect,color]=entrenar_base();

%----------Para usar la base de conocimiento ya entrenada----------
fileID = fopen('C:\Users\Víctor\Documents\MATLAB\IA\vect.txt','r');
formatSpec = '%f';
sizevect = [3 Inf];
vect = fscanf(fileID,formatSpec,sizevect);
fclose(fileID);

fileID = fopen('C:\Users\Víctor\Documents\MATLAB\IA\color.txt','r');
formatSpec = '%f';
sizecolor = [40 3];
color = fscanf(fileID,formatSpec,sizecolor);
fclose(fileID);
%----------------------------------------------------------------------

nuev(1,1)=nuev(1,1)/max(vect(1,:));
nuev(2,1)=nuev(2,1)/max(vect(2,:));
nuev(3,1)=nuev(3,1)/max(vect(3,:));
vect(1,:)=vect(1,:)/max(vect(1,:));
vect(2,:)=vect(2,:)/max(vect(2,:));
vect(3,:)=vect(3,:)/max(vect(3,:));

figure
hold on
for i=1:(length(vect))+1
    if (i <= length(vect)*0.25)
        graf1=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= length(vect)*0.50 && i>length(vect)*0.25)
        graf2=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= length(vect)*0.75 && i>length(vect)*0.50)
        graf3=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= length(vect) && i>length(vect)*0.75)
        graf4=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= length(vect)+1 && i>length(vect))
        graf5=scatter3(nuev(1),nuev(2),nuev(3),25,'b','filled');
    end
end
legend([graf1,graf2,graf3,graf4,graf5],
{'Arandela','Tornillo','Tuerca','Clavo','Nueva imagen'}, 'FontSize',7.5);
xlabel('valor maximo diagonal');
ylabel('área');
zlabel('cuadrado');;
title('Base de conocimiento con imagen a clasificar - KNN');
grid on
hold off

dist=zeros(1,length(vect));
dist_marcas=zeros(length(vect),3);
```

```
for i=1:length(vect)
    dist(1,i)=((nuev(1,1)-vect(1,i))^2+(nuev(2,1)-vect(2,i))^2+(nuev(3,1)-
vect(3,i))^2)^0.5;
    dist_marcas(i,:)=color(i,:);
end
dist;
dist_marcas;
ordenado=zeros(1,length(vect));
ordenado_marcas=zeros(length(vect),3);
aux=0;
aux2=zeros(1,3);
 for j=1:length(vect)-1
    for i=1:length(vect)-1
        if(dist(i)>dist(i+1))
            aux=dist(i);
            dist(i)=dist(i+1);
            dist(i+1)=aux;
            aux2=dist_marcas(i,:);
            dist_marcas(i,:)=dist_marcas(i+1,:);
            dist_marcas(i+1,:)=aux2;
        end
    end
 end
dist;
dist_marcas;
aux=0;

tipo=zeros(1,4);
for i=1:k
    if dist_marcas(i,:)==[0 1 0]
        tipo(1,1)=tipo(1,1)+1;
    end
    if dist_marcas(i,:)==[0 0 0]
        tipo(1,2)=tipo(1,2)+1;
    end
    if dist_marcas(i,:)==[1 0 0]
        tipo(1,3)=tipo(1,3)+1;
    end
    if dist_marcas(i,:)==[0 1 1]
        tipo(1,4)=tipo(1,4)+1;
    end
end

if (tipo(1,1)>tipo(1,2)) && (tipo(1,1)>tipo(1,3)) && (tipo(1,1)>tipo(1,4))
    disp('Es una Arandela');
    color2=[0 1 0];
end
if (tipo(1,2)>tipo(1,1)) && (tipo(1,2)>tipo(1,3)) && (tipo(1,2)>tipo(1,4))
    disp('Es un Tornillo');
    color2=[0 0 0];
end
if (tipo(1,3)>tipo(1,1)) && (tipo(1,3)>tipo(1,2)) && (tipo(1,3)>tipo(1,4))
    disp('Es una Tuerca');
    color2=[1 0 0];
end
if (tipo(1,4)>tipo(1,1)) && (tipo(1,4)>tipo(1,2)) && (tipo(1,4)>tipo(1,3))
```

```
    disp('Es un Clavo');
    color2=[0 1 1];
end
 for j=1:length(tipo)-1
    for i=1:length(tipo)-1
        if(tipo(i)<tipo(i+1))
            aux=tipo(i);
            tipo(i)=tipo(i+1);
            tipo(i+1)=aux;
        end
    end
 end

if tipo(1)==tipo(2)
    disp('Pruebe con otro valor de k');
    error=1;
else
    error=0;
    figure
    hold on
    for i=1:(length(vect))+1
        if (i <= length(vect)*0.25)
            graf1=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= length(vect)*0.50 && i>length(vect)*0.25)
            graf2=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= length(vect)*0.75 && i>length(vect)*0.50)
            graf3=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= length(vect) && i>length(vect)*0.75)
            graf4=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= length(vect)+1 && i>length(vect))
            graf5=scatter3(nuev(1),nuev(2),nuev(3),25,color2,'filled');
        end
    end
    legend([graf1,graf2,graf3,graf4,graf5],
{'Arandela','Tornillo','Tuerca','Clavo','Nueva imagen'}, 'FontSize',7.5);
    xlabel('valor maximo diagonal');
    ylabel('área');
    zlabel('cuadrado');
    title('Algoritmo - KNN');
    grid on
    hold off
end
fprintf('\n')
```

```
function []=Kmeans(nuev)

%----------Para entrenar otra base de conocimiento----------
%[vect,color]=entrenar_base();

%-------------------------------------------------------------------------
fileID = fopen('C:\Users\Víctor\Documents\MATLAB\IA\vect.txt','r');
formatSpec = '%f';
sizevect = [3 Inf];
vect = fscanf(fileID,formatSpec,sizevect);
fclose(fileID);

fileID = fopen('C:\Users\Víctor\Documents\MATLAB\IA\color.txt','r');
formatSpec = '%f';
sizecolor = [40 3];
color = fscanf(fileID,formatSpec,sizecolor);
fclose(fileID);
%-------------------------------------------------------------------------

long=length(vect);
vect(1,long+1)=nuev(1,1);
vect(2,long+1)=nuev(2,1);
vect(3,long+1)=nuev(3,1);

vect(1,:)=vect(1,:)/max(vect(1,:));
vect(2,:)=vect(2,:)/max(vect(2,:));
vect(3,:)=vect(3,:)/max(vect(3,:));

color(long+1,1)=0;
color(long+1,2)=0;
color(long+1,3)=1;


figure
hold on
for i=1:(length(vect))
    if (i <= ((length(vect)-1)*0.25))
        graf1=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= (length(vect)-1)*0.50 && i>(length(vect)-1)*0.25)
        graf2=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= (length(vect)-1)*0.75 && i>(length(vect)-1)*0.50)
        graf3=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= (length(vect)-1) && i>(length(vect)-1)*0.75)
        graf4=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
    end
    if (i <= length(vect) && i>(length(vect)-1))
        graf5=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:),'filled');
    end
end
legend([graf1,graf2,graf3,graf4,graf5],
{'Arandela','Tornillo','Tuerca','Clavo','Nueva imagen'}, 'FontSize',7.5);
xlabel('valor maximo diagonal');
```

```
ylabel('área');
zlabel('cuadrado');
title('Base de conocimiento con imagen a clasificar - k-means')
grid on
hold off


k=4;
delta=0.005;
distintos=0;
it=0;
while(distintos==0)
    columna1=round((10-1)*rand+1);
    columna2=round((20-11)*rand+11);
    columna3=round((30-21)*rand+21);
    columna4=round((40-31)*rand+31);
    if (columna1~=columna2 && columna1~=columna3 && columna1~=columna4)
        if (columna2 ~= columna3 && columna2 ~= columna4)
            if (columna3 ~= columna4)
                if (columna1~=0 && columna2~=0 && columna3~=0 && columna4~=0)
                    distintos=1;
                end
            end
        end
    end
    if it>100
        distintos=1;
    end
    it=it+1;
end
centro1=[vect(1,columna1);vect(2,columna1);vect(3,columna1)];
centro2=[vect(1,columna2);vect(2,columna2);vect(3,columna2)];
centro3=[vect(1,columna3);vect(2,columna3);vect(3,columna3)];
centro4=[vect(1,columna4);vect(2,columna4);vect(3,columna4)];


flag=1;
iter=0;
while(flag==1)
    if(iter==0)
        for i=1:length(vect)
            if (i~=columna1 && i~=columna2 && i~=columna3 && i~=columna4)
                dist1(i)=((centro1(1,1)-vect(1,i))^2+(centro1(2,1)-
vect(2,i))^2+(centro1(3,1)-vect(3,i))^2)^0.5;
                dist2(i)=((centro2(1,1)-vect(1,i))^2+(centro2(2,1)-
vect(2,i))^2+(centro2(3,1)-vect(3,i))^2)^0.5;
                dist3(i)=((centro3(1,1)-vect(1,i))^2+(centro3(2,1)-
vect(2,i))^2+(centro3(3,1)-vect(3,i))^2)^0.5;
                dist4(i)=((centro4(1,1)-vect(1,i))^2+(centro4(2,1)-
vect(2,i))^2+(centro4(3,1)-vect(3,i))^2)^0.5;
            end
        end
    end
    if(iter>0)
        for i=1:length(vect)
```

```
            dist1(i)=((centro1(1,1)-vect(1,i))^2+(centro1(2,1)-
vect(2,i))^2+(centro1(3,1)-vect(3,i))^2)^0.5;
            dist2(i)=((centro2(1,1)-vect(1,i))^2+(centro2(2,1)-
vect(2,i))^2+(centro2(3,1)-vect(3,i))^2)^0.5;
            dist3(i)=((centro3(1,1)-vect(1,i))^2+(centro3(2,1)-
vect(2,i))^2+(centro3(3,1)-vect(3,i))^2)^0.5;
            dist4(i)=((centro4(1,1)-vect(1,i))^2+(centro4(2,1)-
vect(2,i))^2+(centro4(3,1)-vect(3,i))^2)^0.5;
        end
    end
    aux1=0;
    aux2=0;
    aux3=0;
    aux4=0;
    sumx1=0;
    sumx2=0;
    sumx3=0;
    sumx4=0;
    sumy1=0;
    sumy2=0;
    sumy3=0;
    sumy4=0;
    sumz1=0;
    sumz2=0;
    sumz3=0;
    sumz4=0;
    for i=1:length(vect)
        if(dist1(i)<dist2(i) && dist1(i)<dist3(i) && dist1(i)<dist4(i))
            aux1=aux1+1;
            sumx1=vect(1,i)+sumx1;
            sumy1=vect(2,i)+sumy1;
            sumz1=vect(3,i)+sumz1;
            color(i,1)=0;
            color(i,2)=1;
            color(i,3)=0;
        end
        if(dist2(i)<dist1(i) && dist2(i)<dist3(i) && dist2(i)<dist4(i))
            aux2=aux2+1;
            sumx2=vect(1,i)+sumx2;
            sumy2=vect(2,i)+sumy2;
            sumz2=vect(3,i)+sumz2;
            color(i,1)=0;
            color(i,2)=0;
            color(i,3)=0;
        end
        if(dist3(i)<dist1(i) && dist3(i)<dist2(i) && dist3(i)<dist4(i))
            aux3=aux3+1;
            sumx3=vect(1,i)+sumx3;
            sumy3=vect(2,i)+sumy3;
            sumz3=vect(3,i)+sumz3;
            color(i,1)=1;
            color(i,2)=0;
            color(i,3)=0;
        end
        if(dist4(i)<dist1(i) && dist4(i)<dist2(i) && dist4(i)<dist3(i))
            aux4=aux4+1;
            sumx4=vect(1,i)+sumx4;
```

```
            sumy4=vect(2,i)+sumy4;
            sumz4=vect(3,i)+sumz4;
            color(i,1)=0;
            color(i,2)=1;
            color(i,3)=1;
        end
    end
    centro1aux=[(sumx1+centro1(1,1))/(aux1+1); (sumy1+centro1(2,1))/(aux1+1);
(sumz1+centro1(3,1))/(aux1+1)];
    centro2aux=[(sumx2+centro2(1,1))/(aux2+1); (sumy2+centro2(2,1))/(aux2+1);
(sumz2+centro2(3,1))/(aux2+1)];
    centro3aux=[(sumx3+centro3(1,1))/(aux3+1); (sumy3+centro3(2,1))/(aux3+1);
(sumz3+centro3(3,1))/(aux3+1)];
    centro4aux=[(sumx4+centro4(1,1))/(aux4+1); (sumy4+centro4(2,1))/(aux4+1);
(sumz4+centro4(3,1))/(aux4+1)];

    figure
    hold on
    for i=1:(length(vect))
        if (i <= ((length(vect)-1)*0.25))
            graf1=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= (length(vect)-1)*0.50 && i>(length(vect)-1)*0.25)
            graf2=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= (length(vect)-1)*0.75 && i>(length(vect)-1)*0.50)
            graf3=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= (length(vect)-1) && i>(length(vect)-1)*0.75)
            graf4=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:));
        end
        if (i <= length(vect) && i>(length(vect)-1))

graf5=scatter3(vect(1,i),vect(2,i),vect(3,i),25,color(i,:),'filled');
        end
    end
    scatter3(centro1aux(1,1),centro1aux(2,1),centro1aux(3,1),50,'x')
    scatter3(centro2aux(1,1),centro2aux(2,1),centro2aux(3,1),50,'x')
    scatter3(centro3aux(1,1),centro3aux(2,1),centro3aux(3,1),50,'x')
    scatter3(centro4aux(1,1),centro4aux(2,1),centro4aux(3,1),50,'x')
    xlabel('valor maximo diagonal');
    ylabel('área');
    zlabel('cuadrado');
    title('Algoritmo k-means')
    grid on
    hold off

    errorx1=abs(centro1(1,1)-centro1aux(1,1));
    errorx2=abs(centro2(1,1)-centro2aux(1,1));
    errorx3=abs(centro3(1,1)-centro3aux(1,1));
    errorx4=abs(centro4(1,1)-centro4aux(1,1));
    errory1=abs(centro1(2,1)-centro1aux(2,1));
    errory2=abs(centro2(2,1)-centro2aux(2,1));
    errory3=abs(centro3(2,1)-centro3aux(2,1));
    errory4=abs(centro4(2,1)-centro4aux(2,1));
    errorz1=abs(centro1(3,1)-centro1aux(3,1));
```

```
    errorz2=abs(centro2(3,1)-centro2aux(3,1));
    errorz3=abs(centro3(3,1)-centro3aux(3,1));
    errorz4=abs(centro4(3,1)-centro4aux(3,1));
    if(errorx1<delta && errorx2<delta && errorx3<delta && errorx4<delta &&
errory1<delta && errory2<delta && errory3<delta && errory4<delta &&
errorz1<delta && errorz2<delta && errorz3<delta && errorz4<delta )
        flag=0;
    end
    if(iter>20)
        flag=0;
    end

    centro1=centro1aux;
    centro2=centro2aux;
    centro3=centro3aux;
    centro4=centro4aux;
    iter=iter+1;
end

if color(length(vect),:)== [0 1 0]
    disp('Es una Arandela');
end
if color(length(vect),:)== [0 0 0]
    disp('Es un Tornillo');
end
if color(length(vect),:)== [1 0 0]
    disp('Es una Tuerca');
end
if color(length(vect),:)== [0 1 1]
    disp('Es un Clavo');
end
```

Ejemplo de aplicación

A continuación, se realizará un ejemplo de aplicación. Comenzamos con el caso más simple, que la imagen resulte satisfactoria en un principio.

```
Command Window
Se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla
Ingrese el nombre del la imagen a analizar sin el formato: Clavo10
```

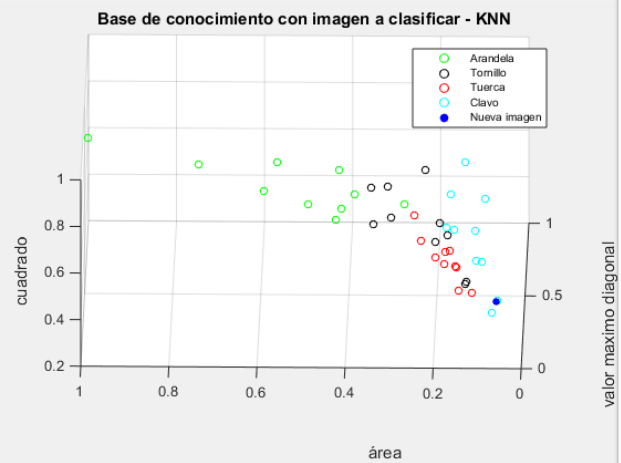
Luego decir que es correcta se correrán los algoritmos. Necesitamos elegir el valor de k de KNN.

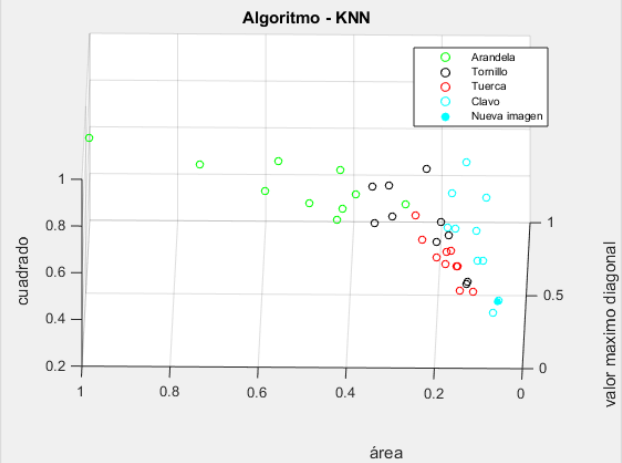
```
Command Window
Se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla
Ingrese el nombre del la imagen a analizar sin el formato: Clavo10
La imagen debe tener fondo de color negro
¿Resulta satisfactoria la imagen segmentada?
(S)Es correcta /(N)No es correcta: s
A continuación se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla

Algoritmo KNN
Ingrese el valor de k del algoritmo KNN:
```

Se puede observar que ambos algoritmos han determinado correctamente la pieza metálica.

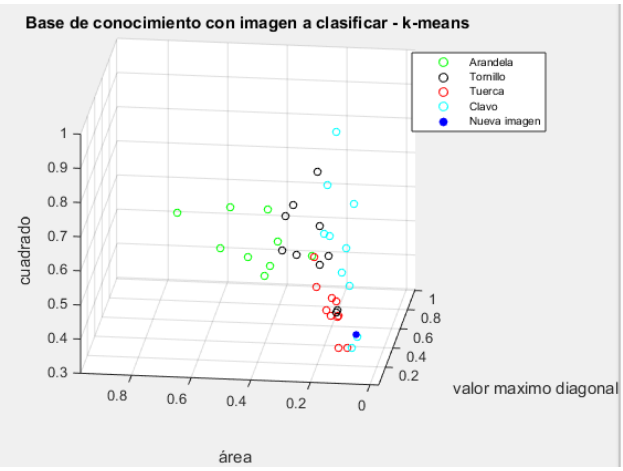
```
Command Window
  Se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla
  Ingrese el nombre del la imagen a analizar sin el formato: Clavo10
  La imagen debe tener fondo de color negro
  ¿Resulta satisfactoria la imagen segmentada?
  (S)Es correcta /(N)No es correcta: s
  A continuación se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla

  Algoritmo KNN
  Ingrese el valor de k del algoritmo KNN: 7
  Es un Clavo

  Algoritmo K-means
  Es un Clavo
fx >>
```

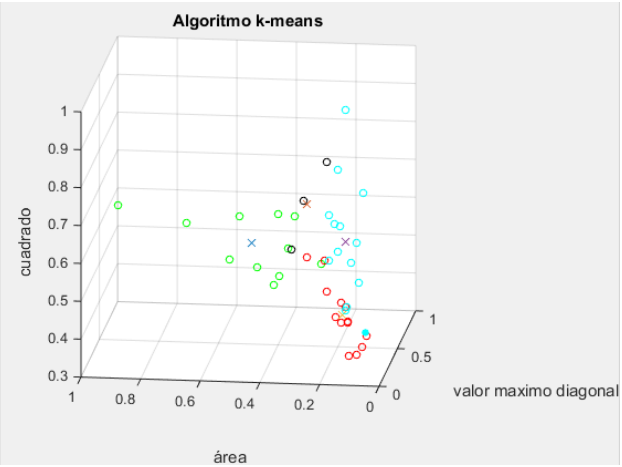
Vemos los diagramas de dispersión de KNN

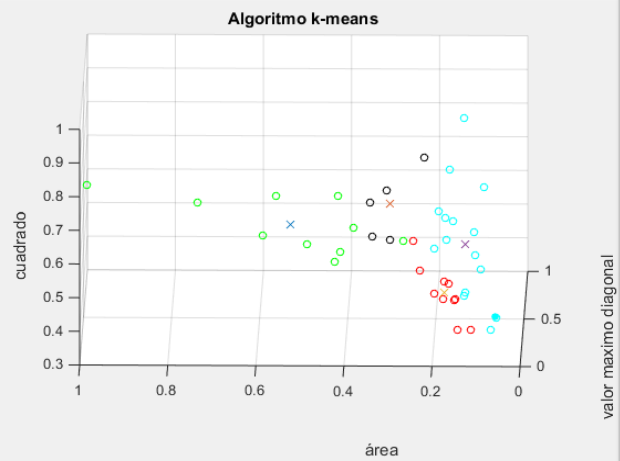


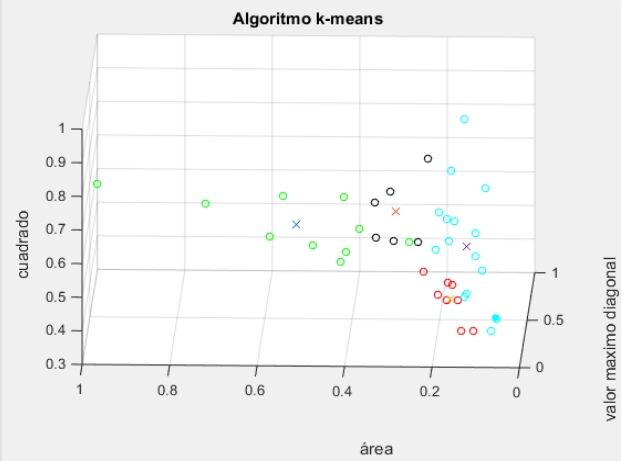


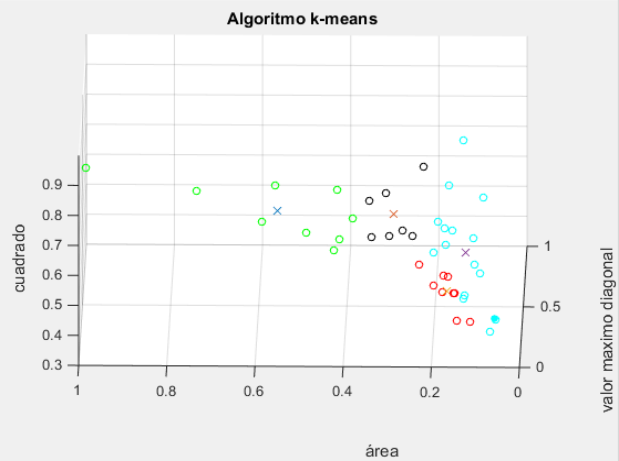
Vemos los diagramas de dispersión de k-means

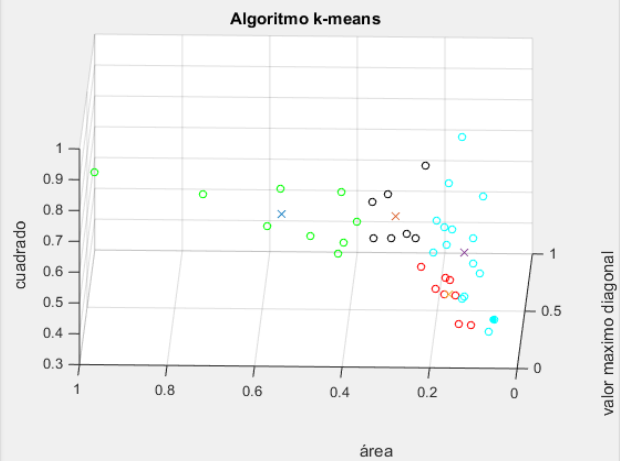





Algoritmo k-means
cuadrado
área
valor maximo diagonal

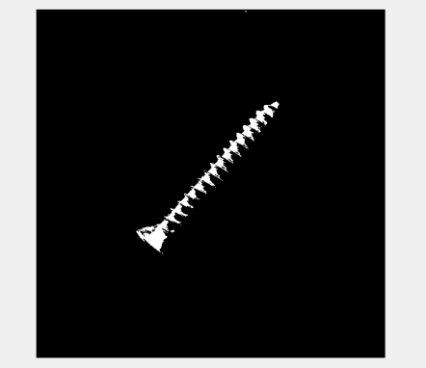

Algoritmo k-means
cuadrado
área
valor maximo diagonal


Algoritmo k-means
cuadrado
área
valor maximo diagonal


Algoritmo k-means
cuadrado
área
valor maximo diagonal

Como segundo caso tenemos que la imagen seleccionada tiene un fondo distinto a los que forman la base de conocimiento, por lo que debemos cambiar alguno de los parámetros para que esta se vea correctamente.

Luego de cambiar estos parámetros me queda de forma correcta y puedo continuar con el programa.

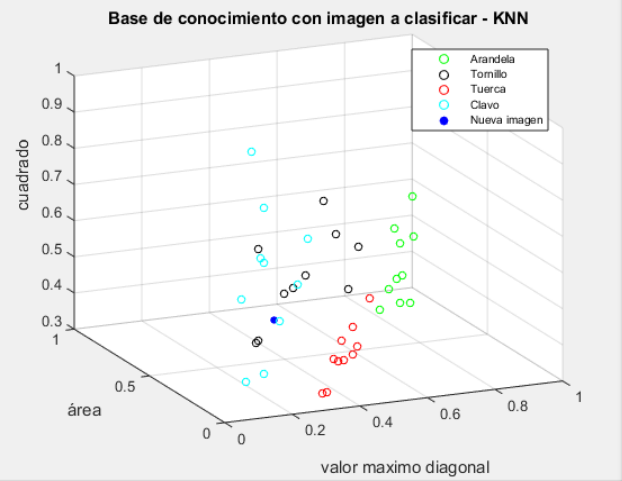


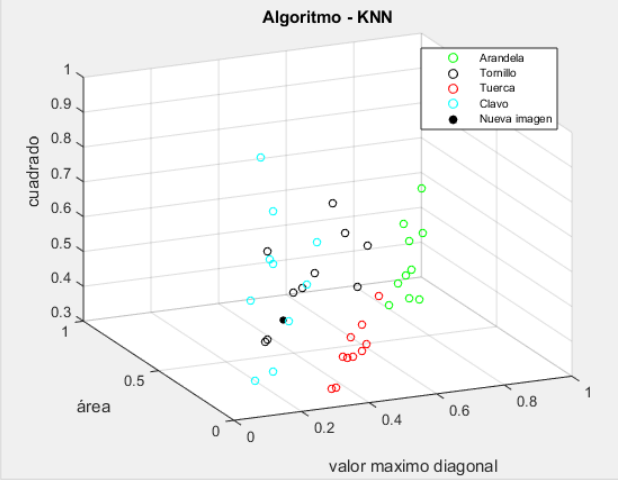
Se puede observar que ambos algoritmos han determinado correctamente la pieza metálica.

Command Window

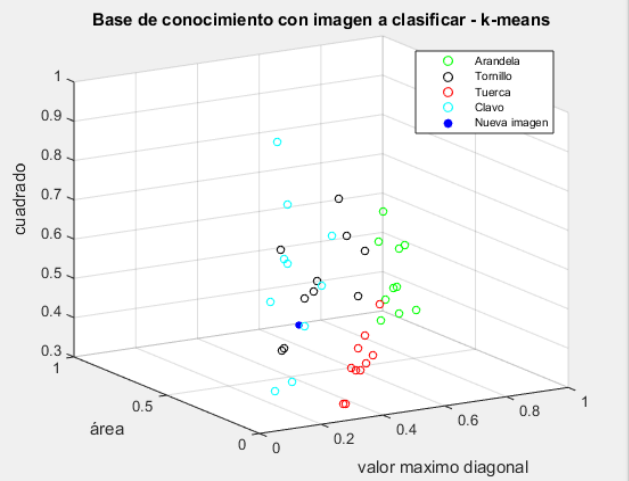
```
Se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla
Ingrese el nombre del la imagen a analizar sin el formato: Tornillo13
La imagen debe tener fondo de color negro
¿Resulta satisfactoria la imagen segmentada?
(S)Es correcta /(N)No es correcta: n
El valor actual de la nivel es de 0.550000
Ingrese un valor para modificar la nivel comprendido entre 0 y 1: 0.40
¿El fondo de la imagén está todo blanco?
(S)Sí /(N)No: s
La imagen debe tener fondo de color negro
¿Resulta satisfactoria la imagen segmentada?
(S)Es correcta /(N)No es correcta: s
A continuación se aplicaran los algoritmos K-means y KNN a la imagen introducida para clasificarla

Algoritmo KNN
Ingrese el valor de k del algoritmo KNN: 7
Es un Tornillo

Algoritmo K-means
Es un Tornillo
```

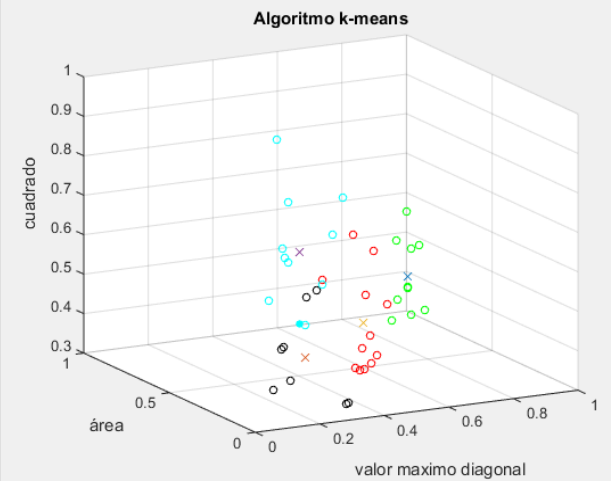
Vemos los diagramas de dispersión de KNN.

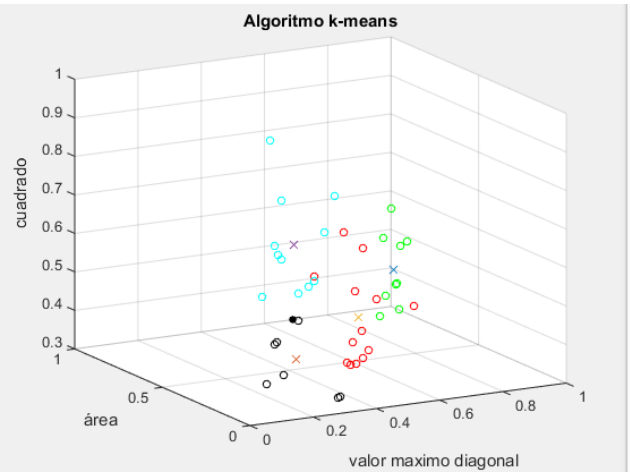


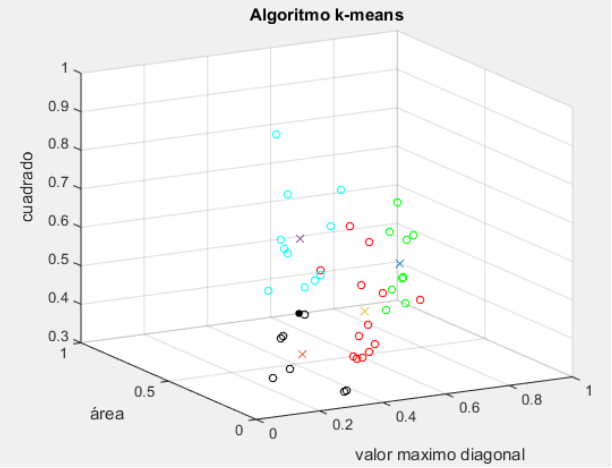


Vemos los diagramas de dispersión de k-means









Resultados

Vamos a analizar 3 arandelas, tornillos, tuercas y clavos adicionales a los que forman la base de conocimiento con fondo similar al usado para la base de conocimiento y uno más de cada uno con un fondo de otro tipo (aparece con +1 en la tabla).

Se puede apreciar en la siguiente tabla la comparación entre el reconocimiento de la imagen por parte de los dos algoritmos:

	KNN		k-menas	
	Correcta	Fallo	Correcta	Fallo
Arandela	3+1	0	3+1	0
Tornillo	1+1	2	2+1	1
Tuerca	3+1	0	3+1	0
Clavo	3+1	0	2	1+1
Total	14	2	13	3

Conclusiones

De acuerdo a los resultados KNN fue efectivo el 87.5% de las veces mientras que k-means lo fue el 81.25% de las veces, si bien ambos tuvieron un alto porcentaje de acierto es preferible utilizar el algoritmo KNN para este tipo de tareas de clasificación debido a que k-means depende de la posición del primer centroide que, aunque se desplace, la pieza a seleccionar puede haber quedado en otro cluster dando resultados erróneos.

Bibliografía utilizada para realizar el trabajo

Apuntes de cátedra

www.mathworks.com

Machine Learning in Action - Peter Harrington

Inteligencia artificial un enfoque moderno – Stuart Russell -2da edición