Experimentos

Autores(Grupo - D):

1. Alejandro, Kenny
2. Gorriz, Oriol
3. Hossain, Tanvir

ÍNDICE

Fase manual

Objetivos

- Evaluar el nivel de precisión de las pruebas con patch manuales.

Recursos

- Patch automatico
- Histogramas de color RB (derivados del modelo RGB)
- Histogramas de color con bins 16 por componente

Collecting manual patches

```
% isExtern = false;
% collectingBoundingbox(isExtern);
% isExtern = true;
```

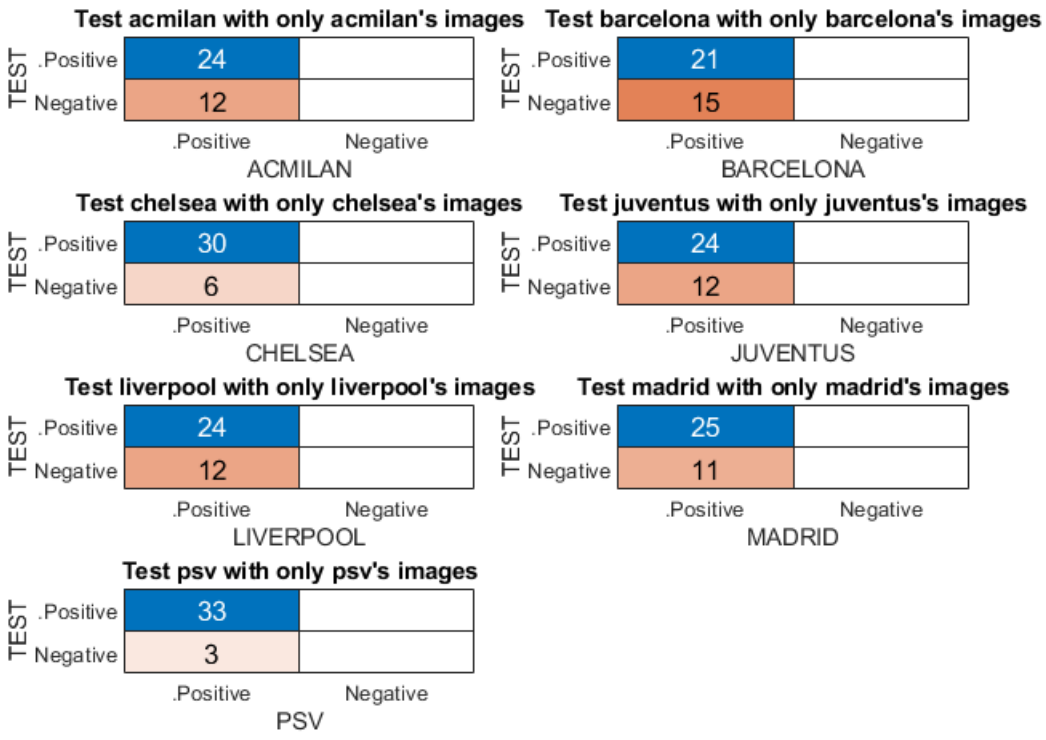
```
% collectingBoundingbox(isExtern);
```

Imagenes internas

- **36 images por equipo**

Test only clubs one club images

```
n_clubs_images = 36;
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_intern_manual.mat")
    for i = 1:7
        range= ([1:n_clubs_images])+ (n_clubs_images * (i-1));
        infoMatrix = infoMatrix_acu(range,:);
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
%         test_number= saveResults("TestsResults_intern_manual_sameClub.mat",confusion_table);
    end
```

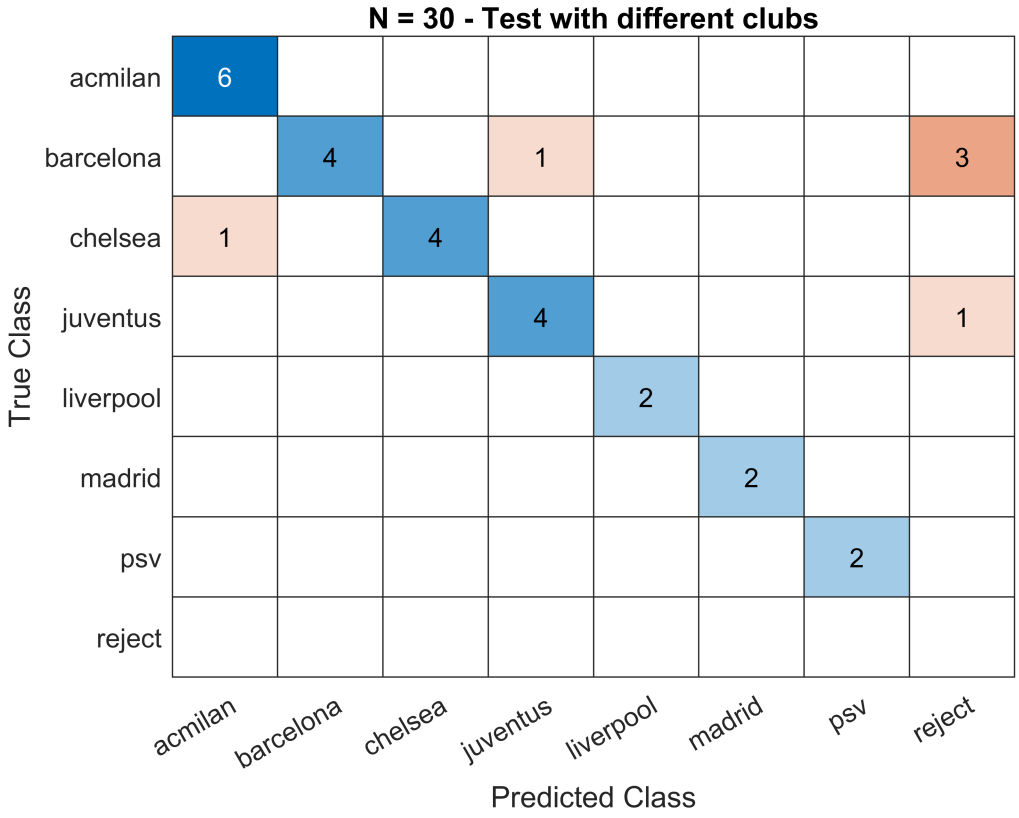


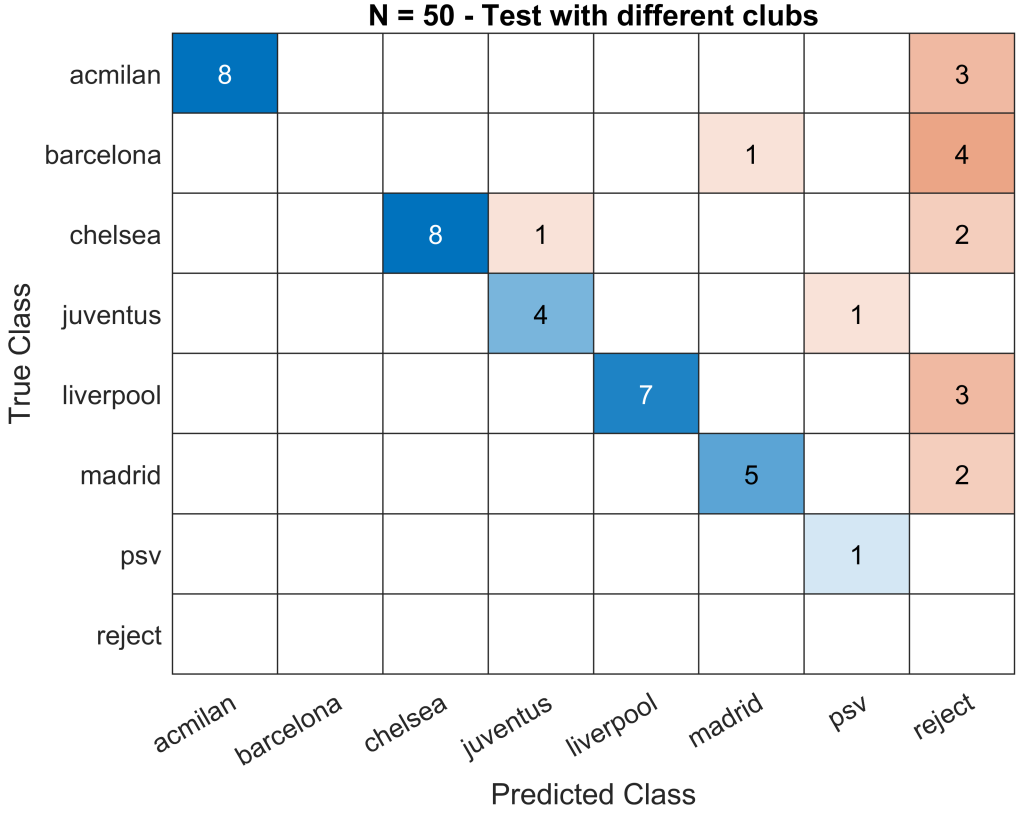

```
catch ME
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isRandom = false;% Means it is not important the second paremeter
```

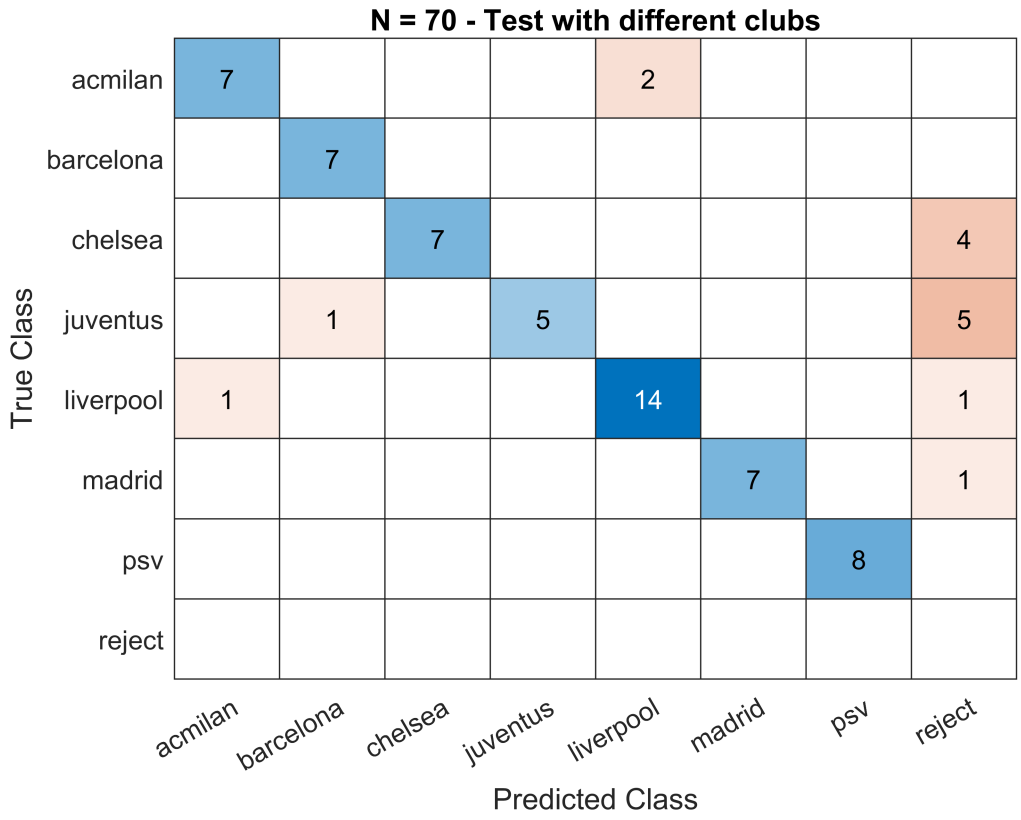
```
    isExtern = false;
    infoMatrix_acu = [];
    isAutomaticPatch = false;
    for i = 1:7
        imgInfo = [ repmat(teams(i),n_clubs_images,1) uint8([1:n_clubs_images]')];
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,0,imgInfo,isAutomaticPatch,isExtern);
        infoMatrix_acu = [infoMatrix_acu;infoMatrix];
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
        test_number= saveResults("TestsResults_intern_manual_sameClub.mat",confusion_table);
    end
    save("infoMatrix_acu_intern_manual.mat","infoMatrix_acu");
end
```

Test multiple clubs

```
try
    load('TestsResults_intern_manual_multipleClubs.mat')
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    n_samples = [30,50,70];
    for i = 1:size(n_samples,2)
        name = sprintf("TestResult_%d",i);
        confusionMatrix = TestsResults.(name);
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
%         saveResults('TestsResults_extern_manual_multipleClubs.mat',confusionMatrix);
    end
```


N = 30 - Test with different clubs
acmilan
barcelona
chelsea
juventus
liverpool
madrid
psv
reject
True Class
Predicted Class
6
4
1
3
1
4
4
1
2
2
2


N = 50 - Test with different clubs
acmilan
barcelona
chelsea
juventus
liverpool
madrid
psv
reject
True Class
Predicted Class
8
3
1
4
8
1
2
4
1
7
3
5
2
1

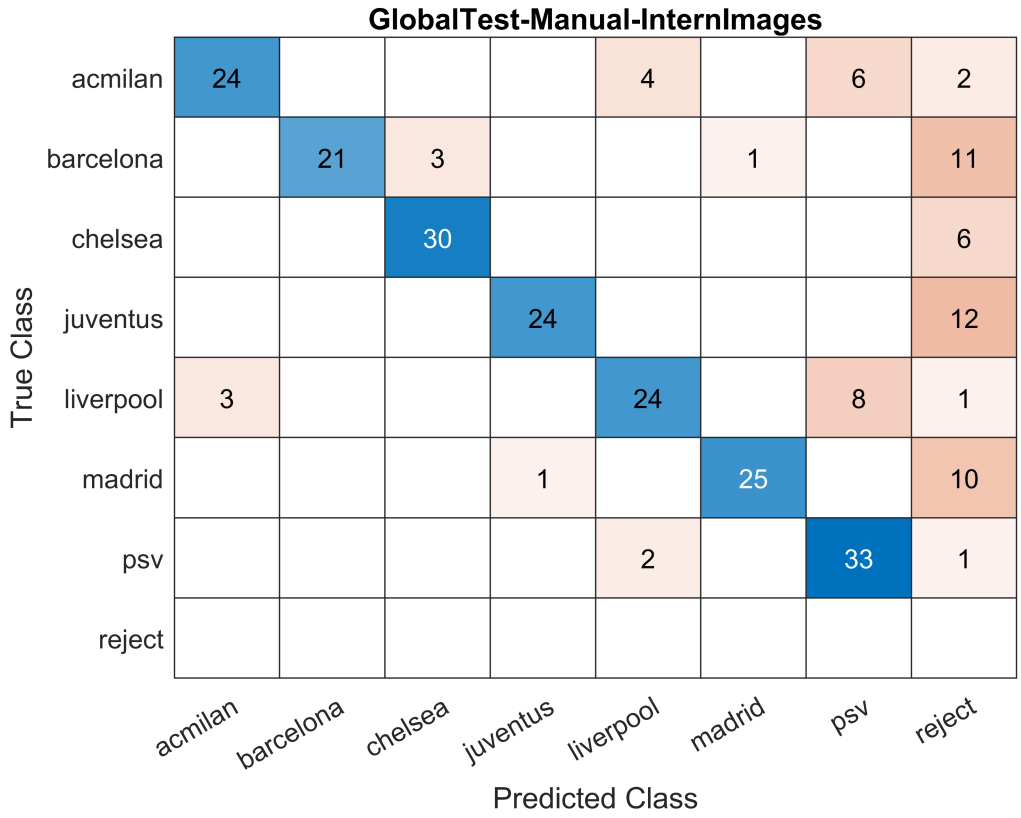



```
catch ME
    %Test and draw
    n_samples = [30,50,70];
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isAutomaticPatch = false;
    isExtern  = true;
    for i = 1:size(n_samples,2)
        isRandom = true;
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,n_samples(i),[],isAutomaticPatch,isExt
        confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
        saveResults('TestsResults_intern_manual_multipleClubs.mat',confusionMatrix);
    end
end
```

Test global

```
teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_intern_manual.mat")
    confusionMatrix = confusionmat(infoMatrix_acu(:,1),infoMatrix_acu(:,3));
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-Manual-InternImages");
```

```
%     saveResults('TestsResults_extern_manual_global.mat',confusionMatrix);
catch ME
    % Directly
    %Test and draw
    isRandom = false;
    n_clubs_images = 36;
    isAutomaticPatch = false;
    isExtern = false;
    img_name = repmat(teams,n_clubs_images,1);
    img_number = repmat([1:n_clubs_images]',1,7);
    infoImg =  [img_name(:),uint8(img_number(:))];
    infoMatrix = testWithMultiplesClubs(isRandom,0,infoImg,isAutomaticPatch,isExtern);
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-Manual-InternImages");
    saveResults('TestsResults_intern_manual_global.mat',confusionMatrix)
end
```



Imagenes externas

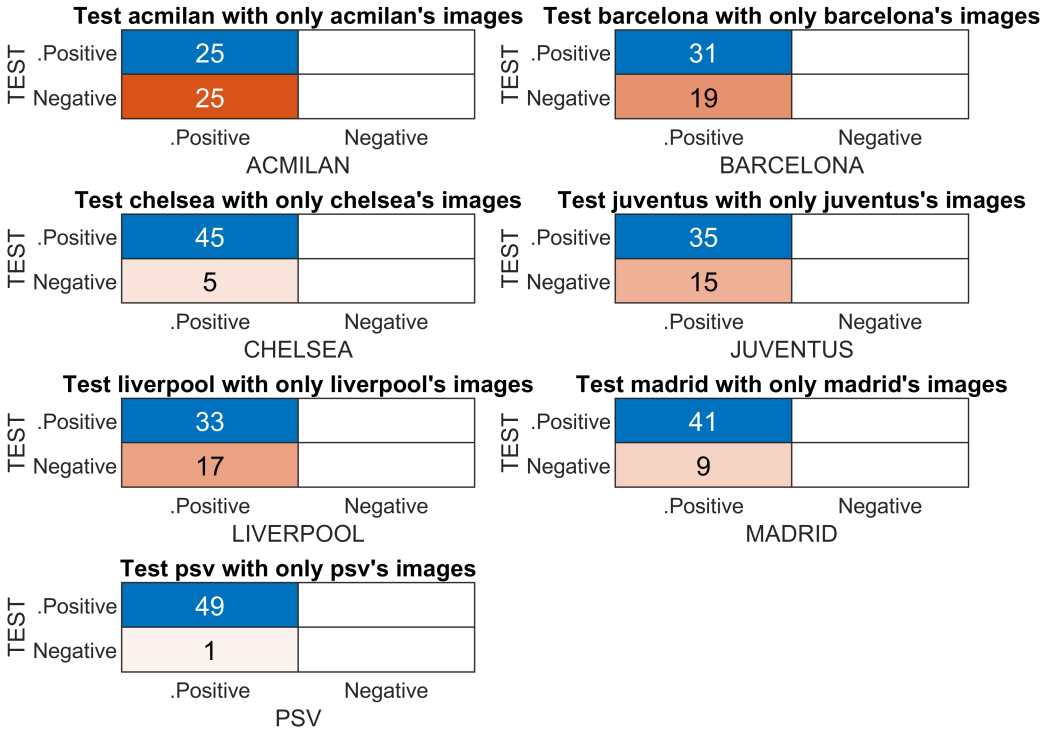
- **50 images por equipo**

```
isExtern = true;
```

Test only clubs one club images

```
n_clubs_images = 50;
try
```

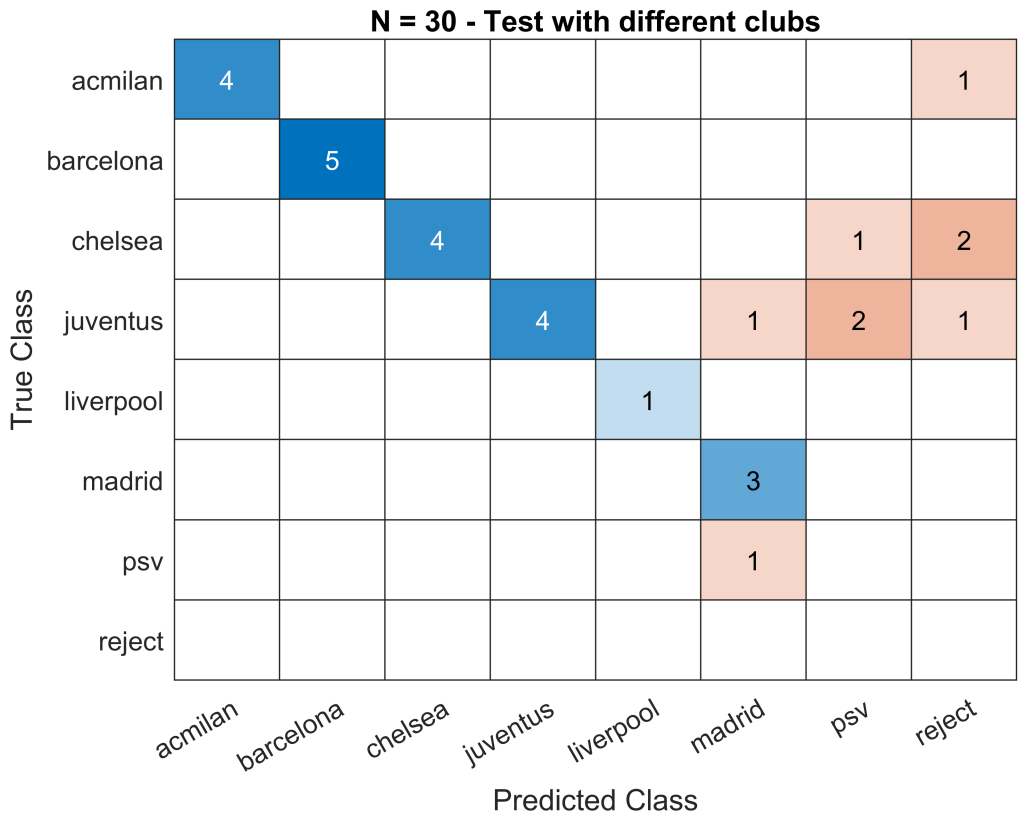
```
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_extern_manual.mat")
    for i = 1:7
        range= ([1:n_clubs_images])+ (n_clubs_images * (i-1));
        infoMatrix = infoMatrix_acu(range,:);
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
%         test_number= saveResults("TestsResults_extern_manual_sameClub.mat",confusion_table);
    end
catch ME
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isRandom = false;% Means it is not important the second paremeter
    isAutomaticPatch = false;
    isExtern = true;
    infoMatrix_acu = [];
    for i = 1:7
        imgInfo = [ repmat(teams(i),n_clubs_images,1) uint8([1:n_clubs_images]')];
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,0,imgInfo,isAutomaticPatch,isExtern);
        infoMatrix_acu = [infoMatrix_acu;infoMatrix];
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
        test_number= saveResults("TestsResults_extern_manual_sameClub.mat",confusion_table)
    end
    save("infoMatrix_acu_extern_manual.mat","infoMatrix_acu");
end
```

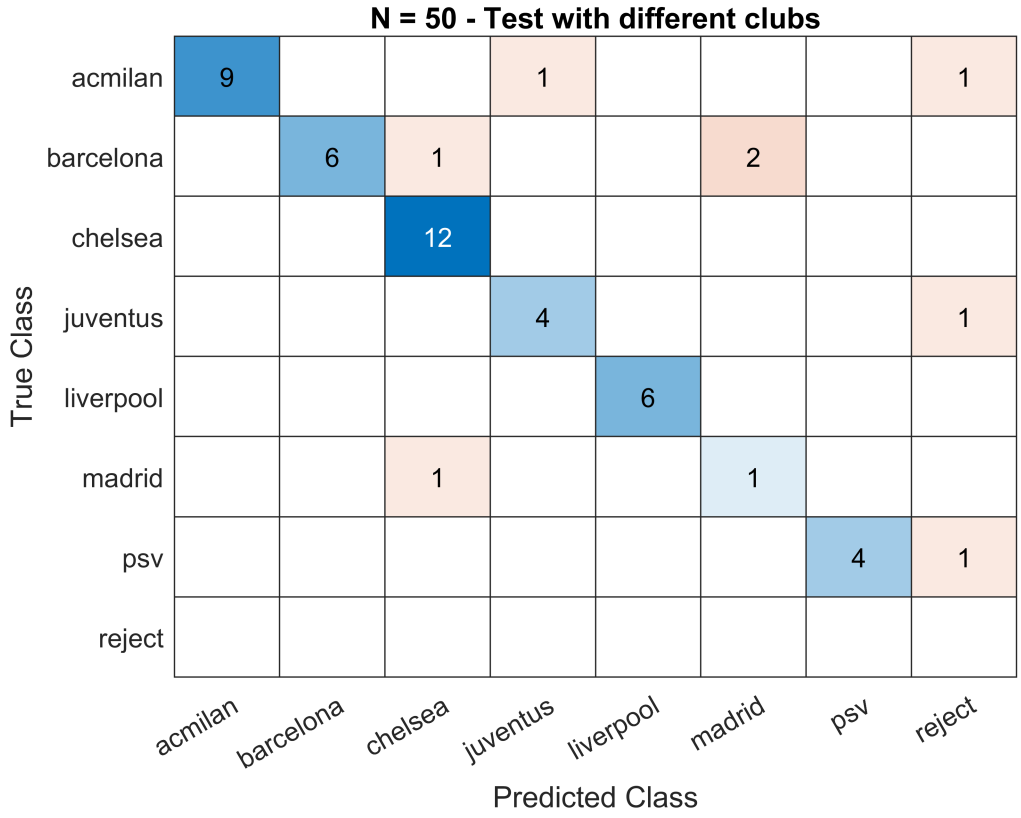


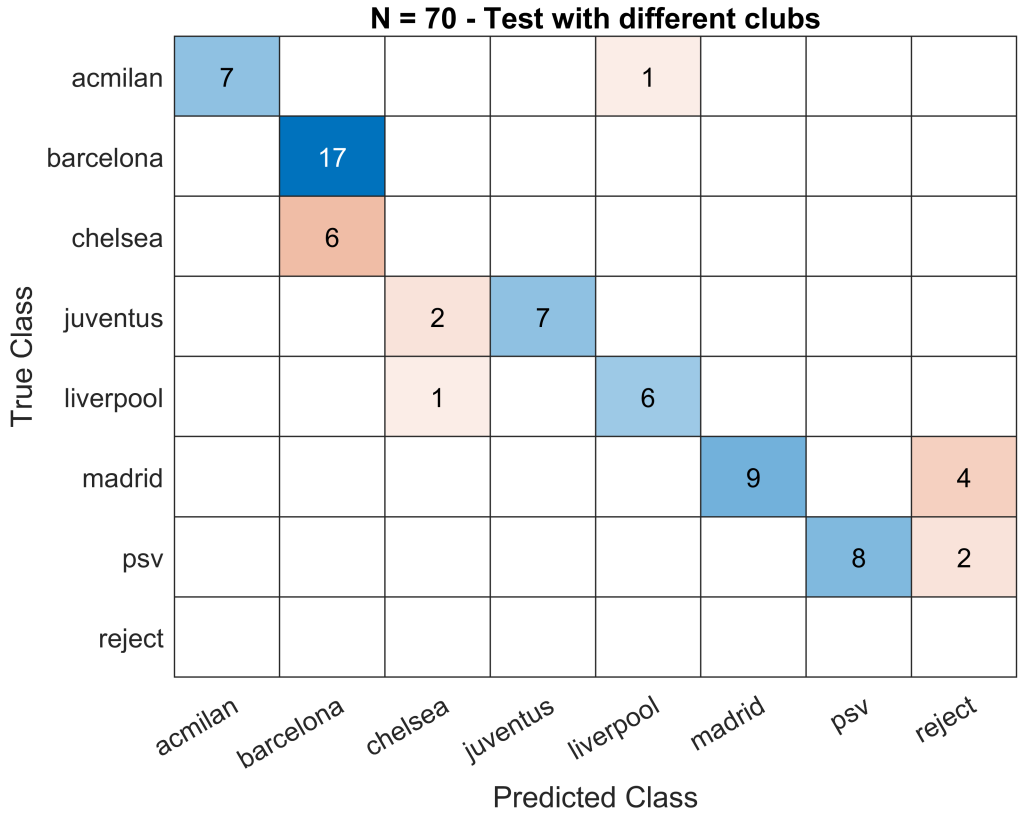
Test multiple clubs

```
%Test and draw
try
    load('TestsResults_extern_manual_multipleClubs.mat')
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    n_samples = [30,50,70];
    for i = 1:size(n_samples,2)
        name = sprintf("TestResult_%d",i);
        confusionMatrix = TestsResults.(name);
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
%         saveResults('TestsResults_extern_manual_multipleClubs.mat',confusionMatrix);
    end
catch ME
    n_samples = [30,50,70];
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isAutomaticPatch = false;
    isExtern  = true;
    for i = 1:size(n_samples,2)
        isRandom = true;
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,n_samples(i),[],isAutomaticPatch,isExt
        confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
```

```
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
        saveResults('TestsResults_extern_manual_multipleClubs.mat',confusionMatrix);
    end
end
```



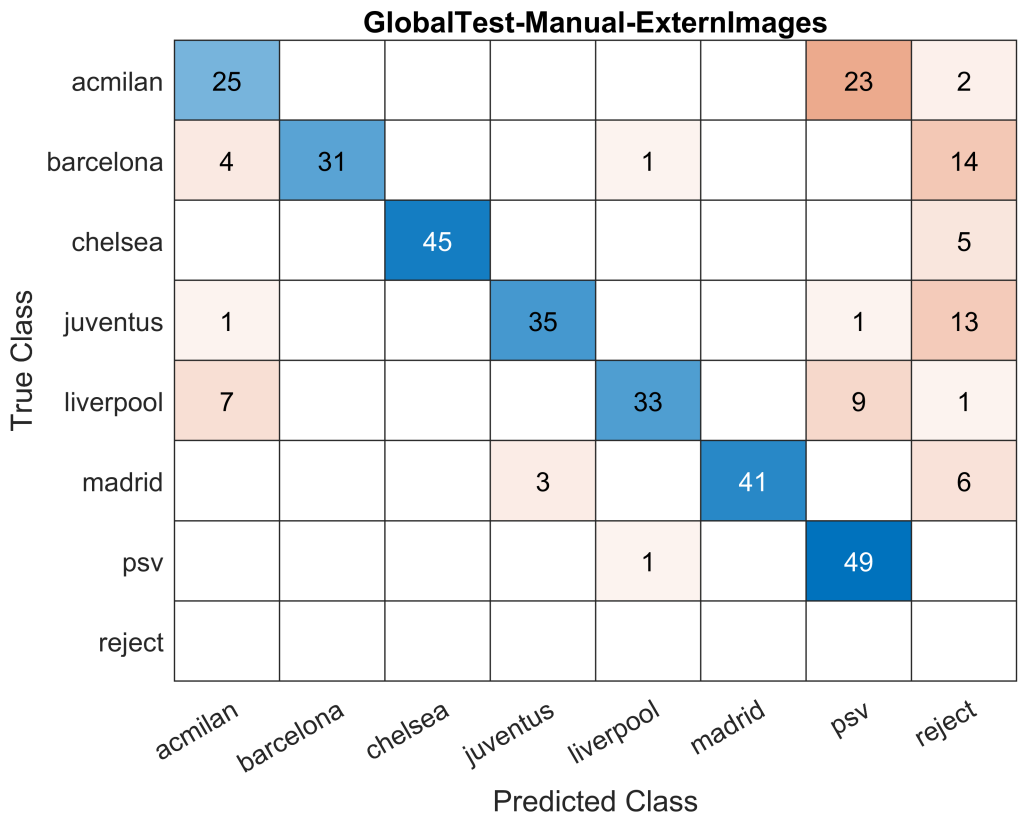




Test global

```
teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
```

```
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_extern_manual.mat")
    confusionMatrix = confusionmat(infoMatrix_acu(:,1),infoMatrix_acu(:,3));
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-Manual-ExternImages");
%     saveResults('TestsResults_extern_manual_global.mat',confusionMatrix);
catch ME
    % Directly
    %Test and draw
    isRandom = false;
    n_clubs_images = 50;
    isAutomaticPatch = false;
    isExtern = true;
    img_name = repmat(teams,n_clubs_images,1);
    img_number = repmat([1:n_clubs_images]',1,7);
    infoImg =  [img_name(:),uint8(img_number(:))];
    infoMatrix = testWithMultiplesClubs(isRandom,0,infoImg,isAutomaticPatch,isExtern);
    confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-Manual-ExternImages");
    saveResults('TestsResults_extern_manual_global.mat',confusionMatrix)
end
```



Fase automatica

Imagenes internas

- **36 images por equipo**

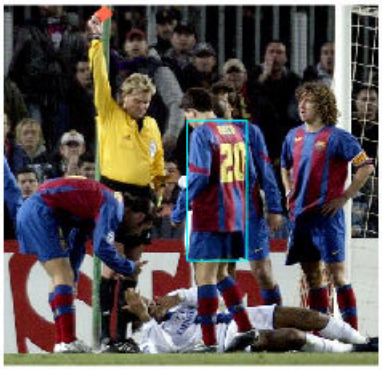
Manual vs automatica

```
isExtern = false;
% Read image
clubs = ["acmilan","barcelona","chelsea", "juventus", "liverpool", "madrid", "psv"];
n_image = 36;
figure
for j = 1:length(clubs)
    club_name = clubs(j);
    file= sprintf('rectangle_intern_%s',club_name);
    load(file);
    for i = 1 %:36
        if(i<10) n = sprintf('%s%d', "0", i);
        else  n = sprintf('%d', i);
        end
        path= sprintf('./imatges_equips/%s/%s.jpg',club_name,n);
        I = imread(path);
        figure
        subplot(1,2,1)
        imshow(I);
        rectangle("Position",rect_box(i,:),"EdgeColor",'cyan');
        title("Manual");
        bb_clubs = bbClub(I);
        % tshirt_box gives start point( "x,y" => "col,row")and the size of the box (width and h
        % tshirt_box = determineShirt();
        if ~isempty(bb_clubs)
            bboxs = [bb_clubs{:,3}];
            [~,index]= sort(bboxs(:,3) * bboxs(:,4),'descend');
            tshirt_box = bboxs(index(1),1:4); % take the bigger one
        else
            %Doing a centered crop of the image
            pct = 80/100; % A percentage of each the size
            tshirt_box = floor([size(I,2)*(1-pct)*0.5 size(I,1)*(1-pct)*0.5 size(I,2)*pct size(
        end
        subplot(1,2,2)
        imshow(I);
        rectangle("Position",tshirt_box,"EdgeColor",'cyan');
        title('Automatic');
    end
end
```

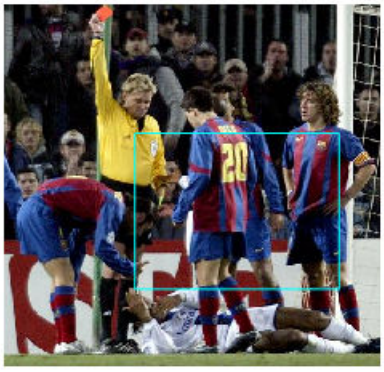
Manual

Automatic

Manual



Automatic



Manual

Automatic

Manual

Automatic

Manual

Automatic

Manual

Automatic

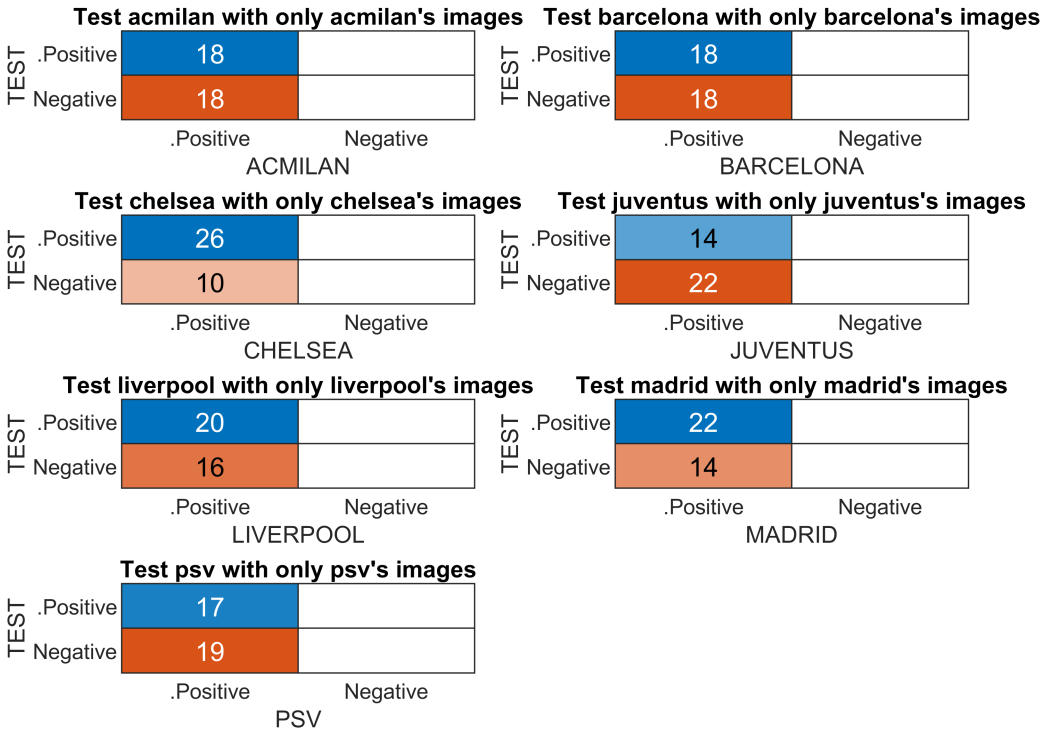
Manual

Automatic

Test only clubs one club images

```
n_clubs_images = 36;
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_intern_automatic.mat")
    for i = 1:7
        range= ([1:n_clubs_images])+ (n_clubs_images * (i-1));
        infoMatrix = infoMatrix_acu(range,:);
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
%         test_number= saveResults("TestsResults_intern_automatic_sameClub.mat",confusion_table
    end
catch ME
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isRandom = false;% Means it is not important the second paremeter
    isExtern = false;
    infoMatrix_acu = [];
    isAutomaticPatch = true;
    for i = 1:7
        imgInfo = [ repmat(teams(i),n_clubs_images,1) uint8([1:n_clubs_images]')];
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,0,imgInfo,isAutomaticPatch,isExtern);
        infoMatrix_acu = [infoMatrix_acu;infoMatrix];
```

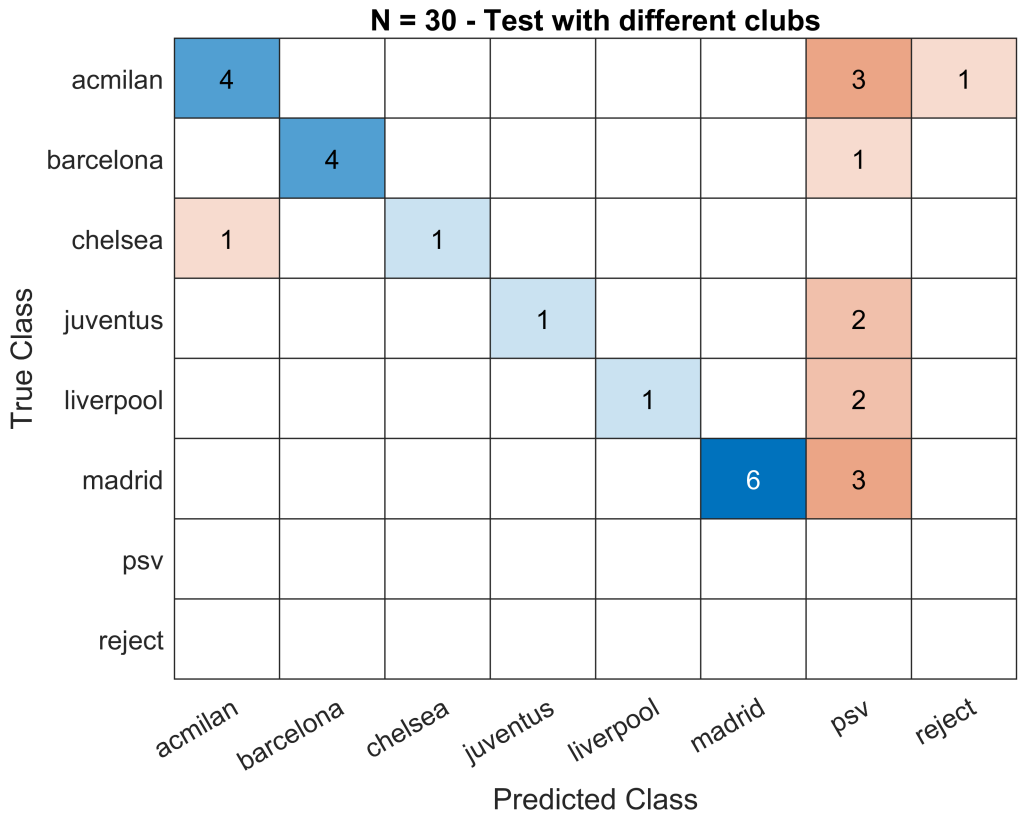
```
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
        test_number= saveResults("TestsResults_intern_automatic_sameClub.mat",confusion_table);
    end
    save("infoMatrix_acu_intern_automatic.mat","infoMatrix_acu");
end
```

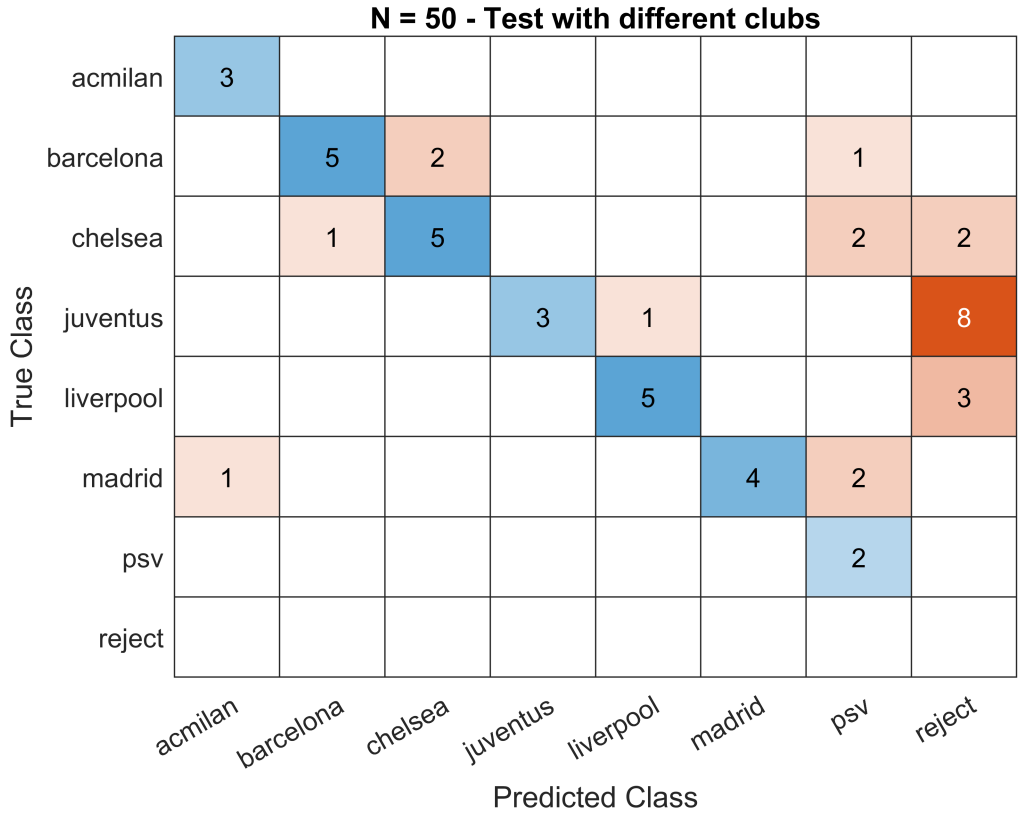


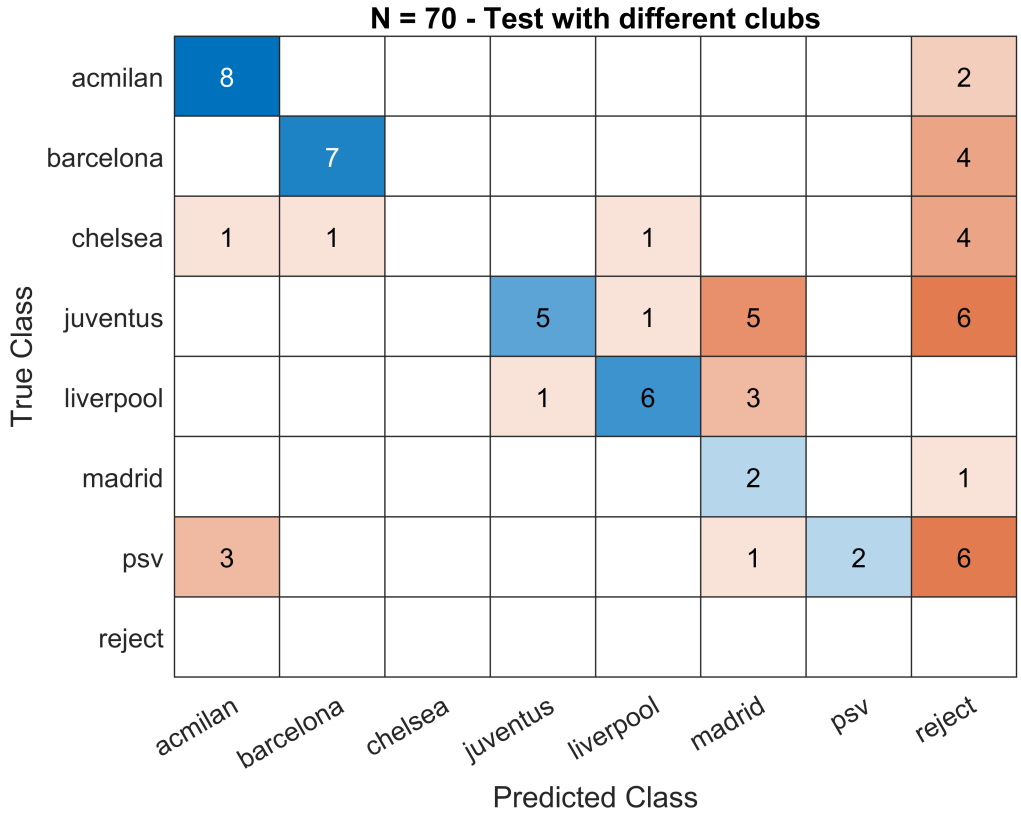
Test multiple clubs

```
%Test and draw
try
    load('TestsResults_intern_automatic_multipleClubs.mat')
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    n_samples = [30,50,70];
    for i = 1:size(n_samples,2)
        name = sprintf("TestResult_%d",i);
        confusionMatrix = TestsResults.(name);
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
%         saveResults('TestsResults_extern_automatic_multipleClubs.mat',confusionMatrix);
    end
catch ME
```

```
    n_samples = [30,50,70];
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isExtern = false;
    isAutomaticPatch = true;
    for i = 1:size(n_samples,2)
        isRandom = true;
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,n_samples(i),[],isAutomaticPatch,isExt
        confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
        saveResults('TestsResults_intern_automatic_multipleClubs.mat',confusionMatrix);
    end
end
```



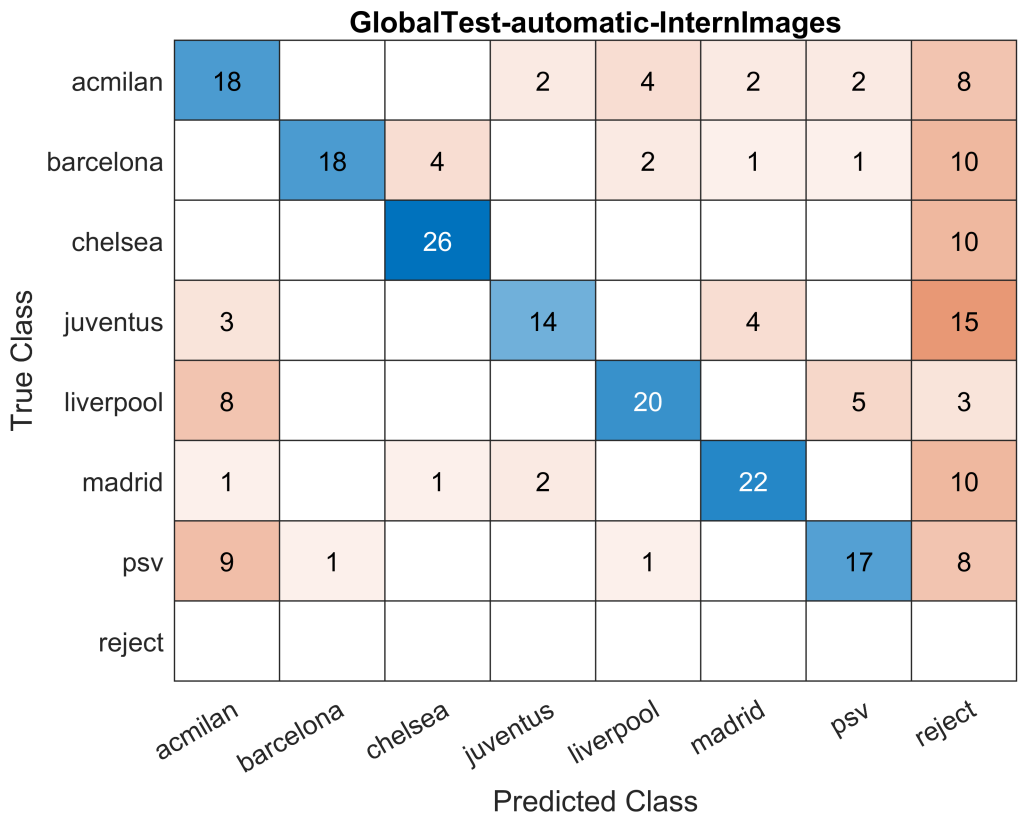




Test global

```
teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
```

```
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_intern_automatic.mat")
    confusionMatrix = confusionmat(infoMatrix_acu(:,1),infoMatrix_acu(:,3));
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-automatic-InternImages");
%     saveResults('TestsResults_extern_automatic_global.mat',confusionMatrix);
catch ME
    % Directly
    %Test and draw
    isRandom = false;
    n_clubs_images = 36;
    isAutomaticPatch = true;
    isExtern = false;
    img_name = repmat(teams,n_clubs_images,1);
    img_number = repmat([1:n_clubs_images]',1,7);
    infoImg =  [img_name(:),uint8(img_number(:))];
    infoMatrix = testWithMultiplesClubs(isRandom,0,infoImg,isAutomaticPatch,isExtern);
    confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-automatic-InternImages");
    saveResults('TestsResults_intern_automatic_global.mat',confusionMatrix)
end
```



Imagenes externas

- **50 images por equipo**

```
isExtern = true;
```

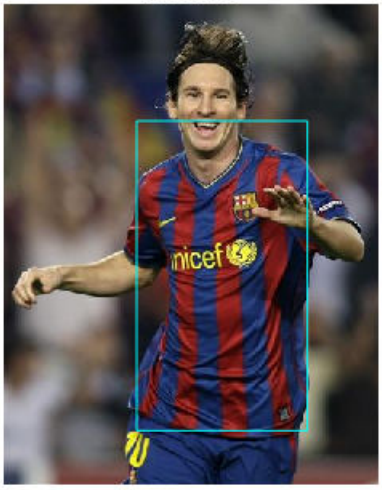
Manual vs automatica

```
% Read image
clubs = ["acmilan","barcelona","chelsea", "juventus", "liverpool", "madrid", "psv"];
n_image = 50;
figure
for j = 1:length(clubs)
    club_name = clubs(j);
    file= sprintf('rectangle_extern_%s',club_name);
    load(file);
%     [selected]
    selected = [3,30,46,9,16,2,45];
%     for i =  %:36
       i = selected(j);
        if(i<10) n = sprintf('%s%d', "0", i);
        else  n = sprintf('%d', i);
        end
        path= sprintf('./testing_equips/%s/%s.jpg',club_name,n);
        I = imread(path);
        figure
        subplot(1,2,1)
        imshow(I);
        rectangle("Position",rect_box(i,:),"EdgeColor",'cyan');
        title("Manual");
        bb_clubs = bbClub(I);
        % tshirt_box gives start point( "x,y" => "col,row")and the size of the box (width and h
        % tshirt_box = determineShirt();
        if ~isempty(bb_clubs)
            bboxs = [bb_clubs{:,3}];
            [~,index]= sort(bboxs(:,3) * bboxs(:,4),'descend');
            tshirt_box = bboxs(index(1),1:4); % take the bigger one
        else
            %Doing a centered crop of the image
            pct = 80/100; % A percentage of each the size
            tshirt_box = floor([size(I,2)*(1-pct)*0.5 size(I,1)*(1-pct)*0.5 size(I,2)*pct size(
        end
        subplot(1,2,2)
        imshow(I);
        rectangle("Position",tshirt_box,"EdgeColor",'cyan');
        title('Automatic');
%     end
end
```

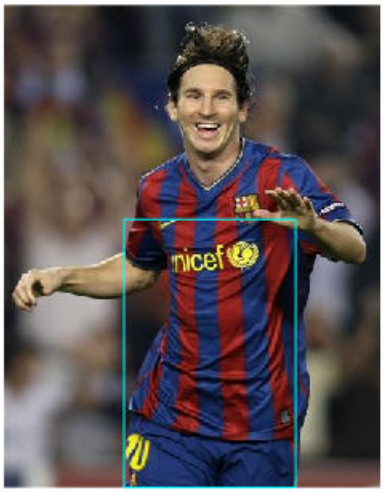
Manual

Automatic

Manual

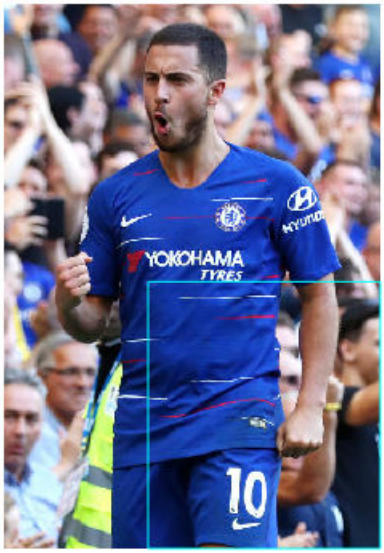


Automatic

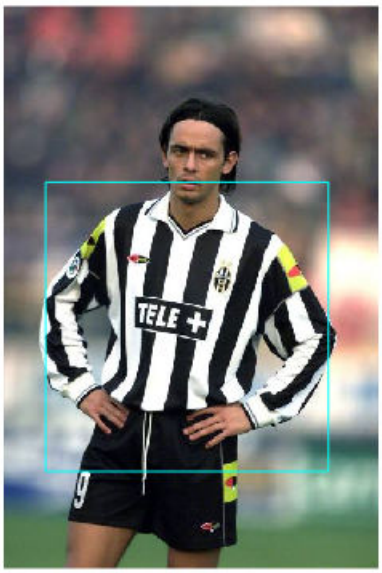


Manual

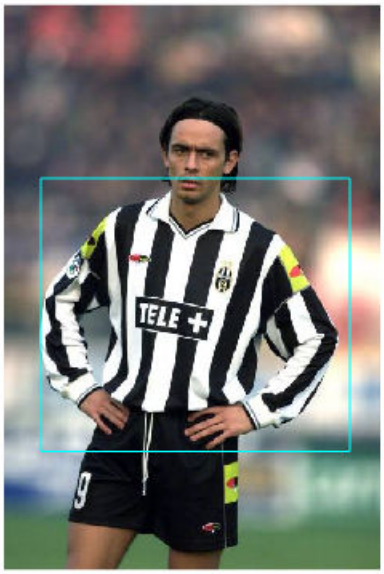
Automatic



Manual



Automatic

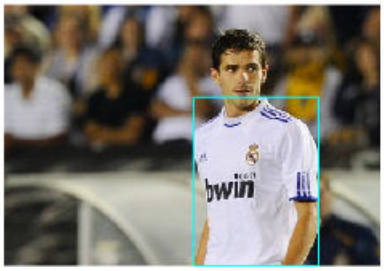


Manual

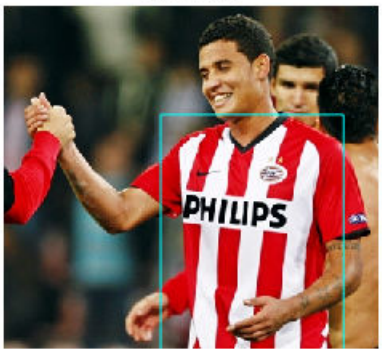
Automatic

Manual

Automatic



Manual

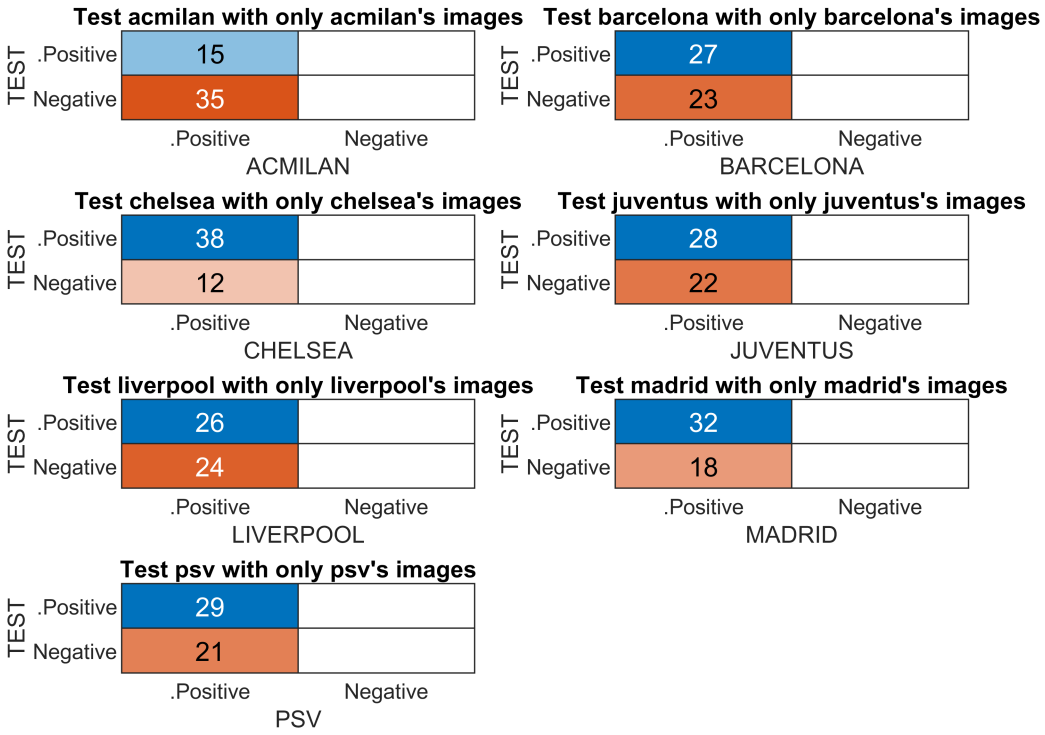


Automatic

Test only clubs one club images

```
n_clubs_images = 50;
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_extern_automatic.mat")
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    figure
    for i = 1:7
        range = ([1:n_clubs_images])+ (n_clubs_images * (i-1));
        infoMatrix = infoMatrix_acu(range,:);
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
%         test_number= saveResults("TestsResults_extern_automatic_sameClub.mat",confusion_table
    end
catch ME
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isRandom = false;% Means it is not important the second paremeter
    isAutomaticPatch = true;
    isExtern = true;
    infoMatrix_acu = [];
    for i = 5:7
        imgInfo = [ repmat(teams(i),n_clubs_images,1) uint8([1:n_clubs_images]')];
```

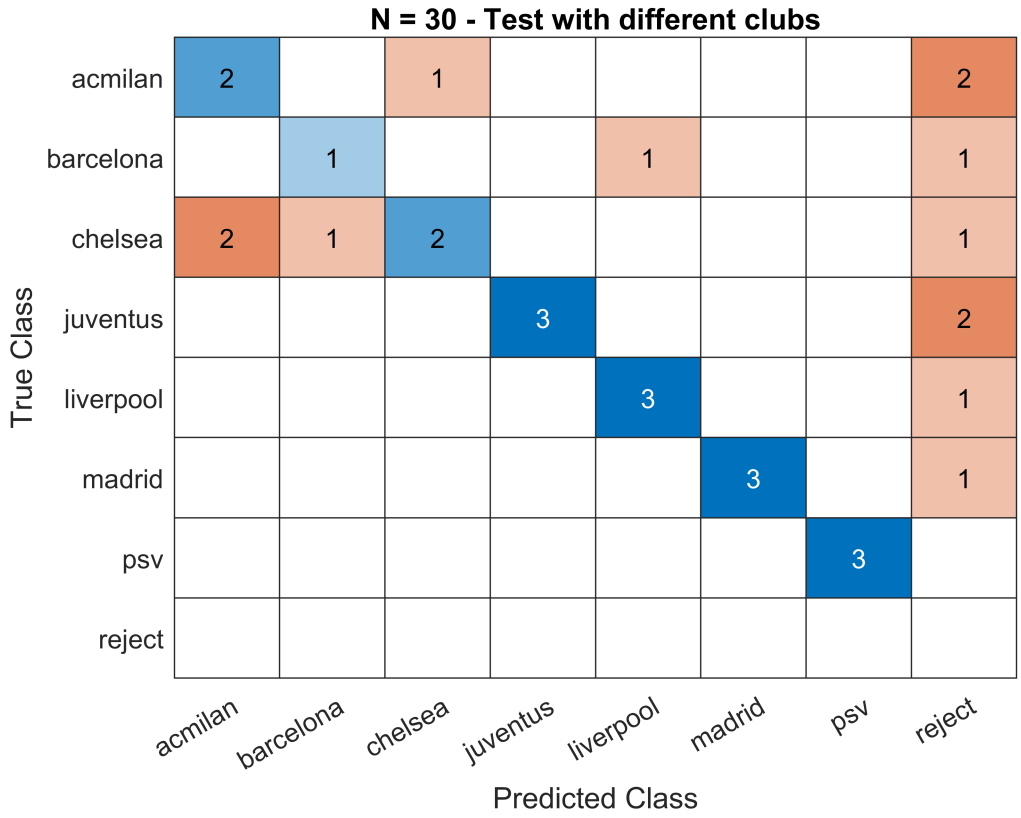
```
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,0,imgInfo,isAutomaticPatch,isExtern);
        infoMatrix_acu = [infoMatrix_acu;infoMatrix];
        confusion_table = buildConfusionTable(infoMatrix,teams(i));
        confusion_table.comment = sprintf("Test %s with only %s's images",teams(i),teams(i));
        confusion_table.n_samples = n_clubs_images;
        subplot(ceil(size(teams,2)/2),2,i)
        drawConfusionTable(confusion_table,teams(i));
        title(confusion_table.comment);
        test_number= saveResults("TestsResults_extern_automatic_sameClub.mat",confusion_table)
    end
    save("infoMatrix_acu_extern_automatic.mat","infoMatrix_acu");
end
```

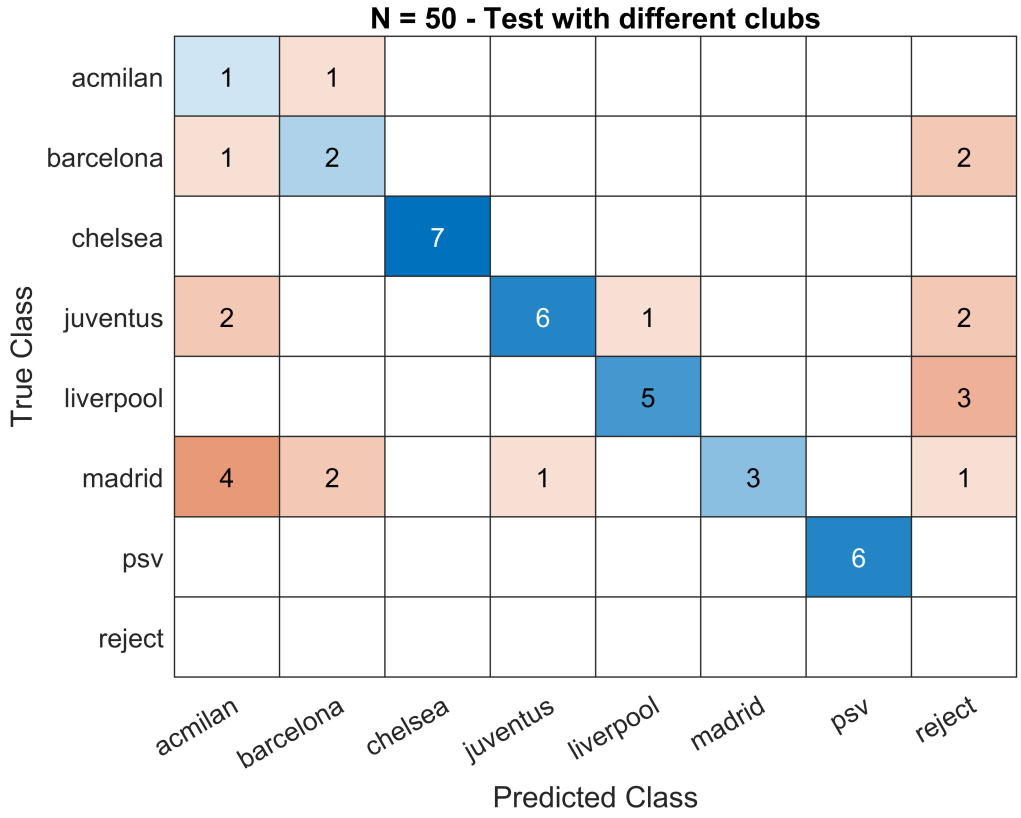


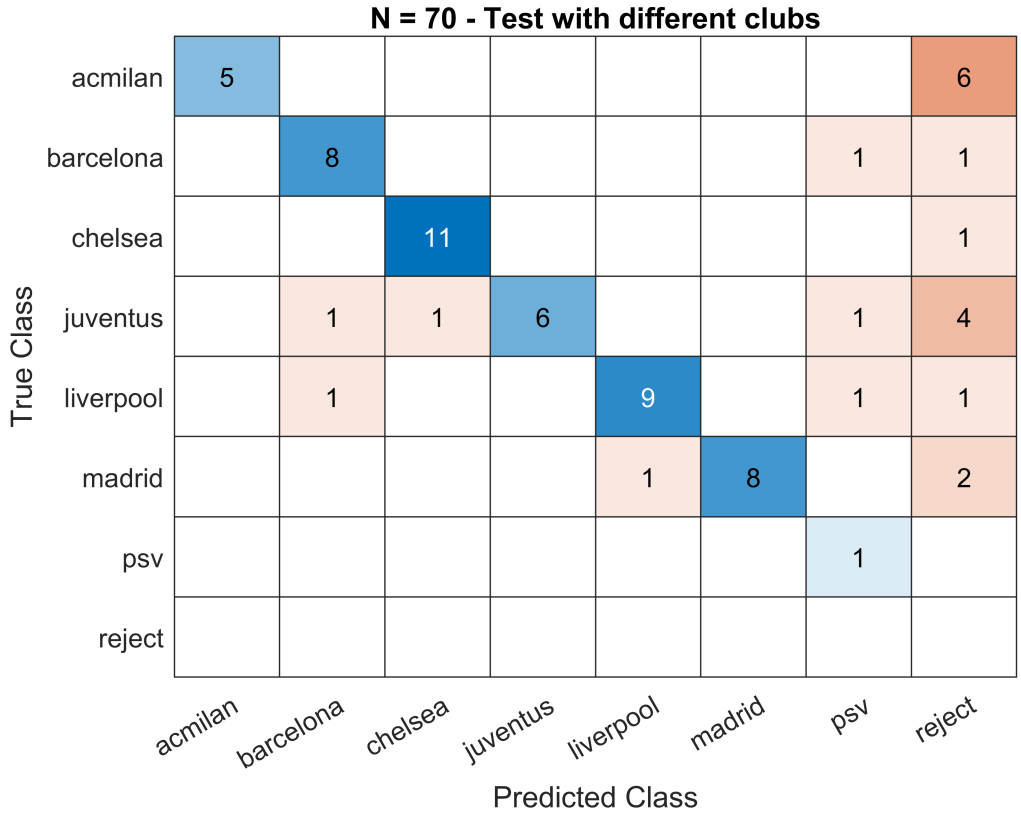
Test multiple clubs

```
%Test and draw
try
    load('TestsResults_extern_automatic_multipleClubs.mat')
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    n_samples = [30,50,70];
    for i = 1:size(n_samples,2)
        name = sprintf("TestResult_%d",i);
        confusionMatrix = TestsResults.(name);
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
%         saveResults('TestsResults_extern_automatic_multipleClubs.mat',confusionMatrix);
```

```
    end
catch ME
    n_samples = [30,50,70];
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    isAutomaticPatch = true;
    isExtern  = true;
    for i = 1:size(n_samples,2)
        isRandom = true;
        [infoMatrix,~] = testWithMultiplesClubs(isRandom,n_samples(i),[],isAutomaticPatch,isExt
        confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
        figure
        confusionchart(confusionMatrix,[teams,"reject"]);
        text = sprintf('N = %d - %s',n_samples(i),"Test with different clubs");
        title(text);
        saveResults('TestsResults_extern_automatic_multipleClubs.mat',confusionMatrix);
    end
end
```



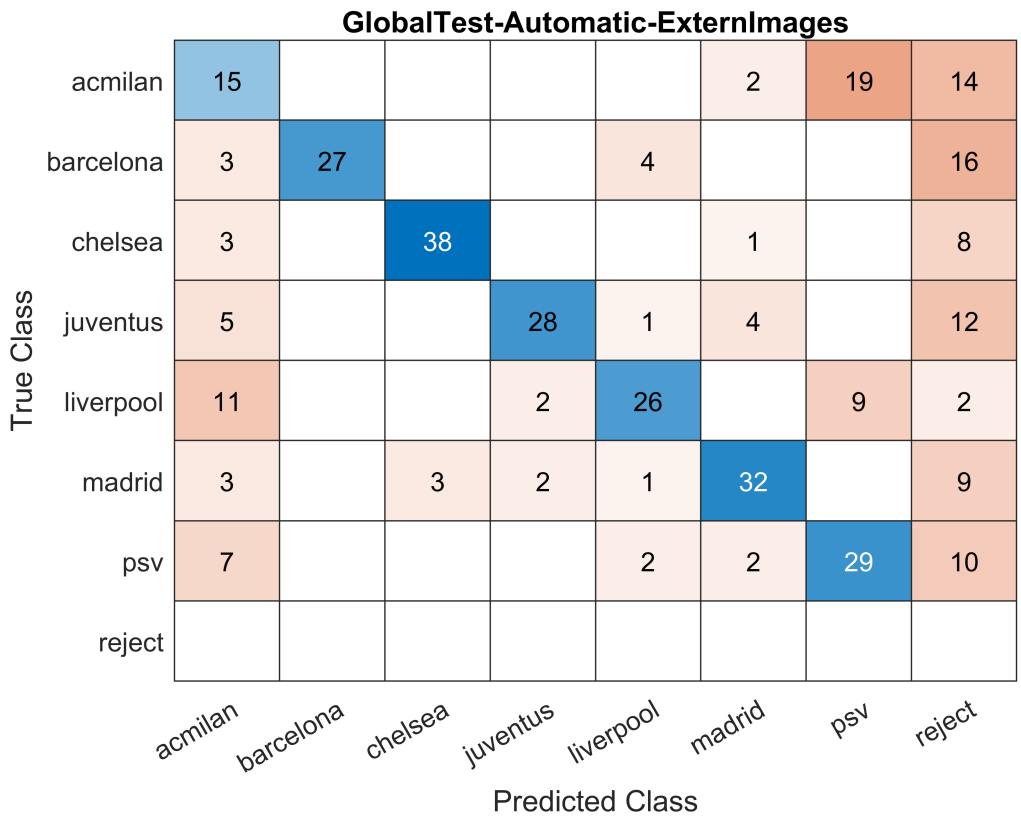




Test global

```
teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
```

```
try
    %try to load the results of infoMatrix_acu previously calculated
    load("infoMatrix_acu_extern_automatic.mat")
    confusionMatrix = confusionmat(infoMatrix_acu(:,1),infoMatrix_acu(:,3));
    figure
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-Automatic-ExternImages");
%     saveResults('TestsResults_extern_automatic_global.mat',confusionMatrix);
catch ME
    % Directly
    %Test and draw
    isRandom = false;
    n_clubs_images = 50;
    isAutomaticPatch = true;
    isExtern = true;
    img_name = repmat(teams,n_clubs_images,1);
    img_number = repmat([1:n_clubs_images]',1,7);
    infoImg =  [img_name(:),uint8(img_number(:))];
    infoMatrix = testWithMultiplesClubs(isRandom,0,infoImg,isAutomaticPatch,isExtern);
    figure
    confusionMatrix = confusionmat(infoMatrix(:,1),infoMatrix(:,3));
    confusionchart(confusionMatrix,[teams,"reject"]);
    title("GlobalTest-Automatic-ExternImages");
    saveResults('TestsResults_extern_automatic_global.mat',confusionMatrix)
end
```



Funciones

Funciones para test

```
function [infoMatrix,res] =  testWithMultiplesClubs(isRandom,numberTests,imgInfo,isAutomaticPat
     %Input:
     % - isRandom indicates if the images are given by the user
     % as "imgInfo" or created randomly.
     % - "imgInfo" is an matrix two columns: "names" and "numbers". It is only considered when
     %   isRandom is false.
     % - numberTests specifys the number of tests. It is only considered when
     %   isRandom is true.
     % Output:
     % -infomatrix with three columns: nameclub numimage and predictedClubName
    teams = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    n_equips = 7;
    n_img_test = numberTests;
    if ~isRandom
        n_img_test = size(imgInfo,1);
    end
    if isExtern
        n_imgs_available = 50;
        name_file = 'testing_equips';
        name_rect = 'rectangle_extern_';
    else
        n_imgs_available = 36;
        name_file = 'imatges_equips';
        name_rect = 'rectangle_intern_';
    end
    Club = [];
    Number = [];
    predictedClubName = [];
    res.additional_data = [];
    res.additional_data.good_sample = [];
    res.additional_data.bad_sample = [];
    for i = 1:n_img_test
        if (isRandom)
           clubName = teams(min(round(1 + rand *(n_equips)),n_equips));
           number = min(round(1 + rand *(n_imgs_available-1)),n_imgs_available);
        else
           clubName = imgInfo(i,1);
           number = str2double(imgInfo(i,2));
        end
        Club = [Club ,clubName];
        Number = [Number,number];
        if(number<10)
            i_tmp = sprintf('%s%d', "0", number);
        else
            i_tmp = sprintf('%d', number);
        end
        path = sprintf('./%s/%s/%s.jpg',name_file,clubName,i_tmp);
        I = imread(path);
        if ~isAutomaticPatch
            %Read a precalculated rectagle
            filename = sprintf("%s%s",name_rect,clubName);
```

```
            load(filename) % load rect_box
            tshirt_box = uint16(rect_box(number,:));
            %Get a rectangle using getrect
            % figure
            % imshow(I);
            % tshirt_box = uint16(getrect);
        else
            tshirt_box = [];
        end
        [predictedClub,additional_data] = clubsClassifier(I,isAutomaticPatch,tshirt_box); % RGB
%         [results] = testWithAClub(selectedTeam,[number],isAutomaticPatch,isExtern);
        predictedClubName = [predictedClubName,predictedClub];
        aux = [];
        aux = additional_data.good_sample;
        res.additional_data.good_sample = [res.additional_data.good_sample; aux];
        aux = [];
        aux = additional_data.bad_sample;
        res.additional_data.bad_sample  = [res.additional_data.bad_sample; aux];

    end
    infoMatrix(:,1) = Club;
    infoMatrix(:,2) = string(Number);
    infoMatrix(:,3) = predictedClubName;
end

function drawConfusionTable(confusion_table,referenceClub)
    m = [confusion_table.TP.counter, confusion_table.FP.counter;
         confusion_table.FN.counter, confusion_table.TN.counter];
    confusionchart(m,[".Positive","Negative"]');
    ylabel('TEST')
    xlabel(upper(referenceClub))
end

function [confusion_table] = buildConfusionTable(infoMatrix,referenceClub)
       %%Confusion Table  o contingencia
        %                              Barcelona
        %                    positivo             negativo
        %   T  positivo    truePositive  1   falsePositive  3
        %   E
        %   S  negativo    falseNegative 2   trueNegative   4
        %   T
        %     1 -> truepositve    TP
        %     2 -> falseNegative  FN
        %     3 -> falsePositive  FP
        %     4 -> trueNegative   TN
        confusion_table.TP.imgInfo = [];
        confusion_table.FN.imgInfo = [];
        confusion_table.FP.imgInfo = [];
        confusion_table.TN.imgInfo = [];
        confusion_table.TP.counter = 0;
        confusion_table.FN.counter = 0;
        confusion_table.FP.counter = 0;
        confusion_table.TN.counter = 0;
        for i = 1:size(infoMatrix,1)
```

```
            isPositive = (infoMatrix(i,1) == infoMatrix(i,3));
            if (infoMatrix(i,1) == referenceClub)
                %Input: ReferenceClub
                %Desition:
                if isPositive
                  confusion_table.TP.counter = confusion_table.TP.counter + 1;%TP
                  confusion_table.TP.imgInfo = [confusion_table.TP.imgInfo;infoMatrix(i,1:2)];
                else
                  confusion_table.FN.counter  = confusion_table.FN.counter + 1;%FN
                  confusion_table.FN.imgInfo = [confusion_table.FN.imgInfo;infoMatrix(i,1:2)];
                end
            else
                %Input: No referenceClub
                %Desition:
                if isPositive
                  confusion_table.FP.counter  =  confusion_table.FP.counter + 1;%FP
                   confusion_table.FP.imgInfo = [confusion_table.FP.imgInfo;infoMatrix(i,1:2)];

                else
                   confusion_table.TN.counter  =  confusion_table.TN.counter + 1;%TN
                   confusion_table.TN.imgInfo = [confusion_table.TN.imgInfo;infoMatrix(i,1:2)];
                end
            end
        end
end

function test_number = saveResults(filename,new_confusion_table)
    try
        load(filename)
        max_num = length(cellfun('length',struct2cell(TestsResults)));
    catch ME
        TestsResults = [];
        max_num = 0;
    end
    test_number = max_num+1;
    custom_name_test = sprintf('TestResult_%d',test_number);
    TestsResults.(custom_name_test) = new_confusion_table;
    save(filename,"TestsResults");
end

function [] = recollectionBoundingbox(isExtern)
    if isExtern
        n_clubs_images = 50;
        name_file = 'testing_equips';
        type = 'extern';
    else
        n_clubs_images = 36;
        name_file = 'imatges_equips';
        type = 'intern';
    end
    clubs = ["acmilan", "barcelona","chelsea", "juventus", "liverpool", "madrid", "psv"];
    for j = 1:7
        club_name = clubs(j);
        rect_box = [];
```

```
        for i=1:n_clubs_images
            if(i<10) n = sprintf('%s%d', "0", i);
            else  n = sprintf('%d', i);
            end
            path = sprintf('./%s/%s/%s.jpg',name_file,club_name,n);
            IModel = imread(path);
            figure
            imshow(IModel);
            rect_box(i,:) = uint32(getrect);
        end
        filename = sprintf("rectangle_%s_%s.mat",type,club_name);
        save(filename,"rect_box");
    end
end
```

Funciones principales

```
function [predictedClubName,additional_data] = clubsClassifier(I,isAutomaticPatch,tshirt_box)
    if ~isAutomaticPatch
        if isempty(tshirt_box) || ~isequal(size(tshirt_box),[1 4])
             error('Pass a valid rectangle. \n Notice:A rectangle is [x y width height]');
        end
    else
        % isAutomaticPatch
        if ~isempty(find(size(I) > 800))
            % if one of the image dimension is higher that 800 therefore we
            % resize the image resizeing the biggest dimension to 1000(the others will also
            %  resize in the same amount)
            maxim = max(size(I));
            ratio = 1/(maxim/800);
            I = imresize(I,ratio);
        end
        tshirt_box = [];
        % Looking for the bounding box or also called patch
        bb_clubs = bbClub(I);
        % tshirt_box gives start point( "x,y" => "col,row")and the size of the box (width and h
        % tshirt_box = determineShirt();
        if ~isempty(bb_clubs)
            bboxs = [bb_clubs{:,3}];
            [~,index]= sort(bboxs(:,3) * bboxs(:,4),'descend');
            tshirt_box = bboxs(index(1),1:4); % take the bigger one
        else
            %Doing a centered crop of the image
            pct = 80/100; % A percentage of each the size
            tshirt_box = floor([size(I,2)*(1-pct)*0.5 size(I,1)*(1-pct)*0.5 size(I,2)*pct size(
        end
    end
    clubs = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
    additional_data.good_sample = [];
    additional_data.bad_sample = [];
    additional_data.exists = [];
        isClubDesition = zeros(1,7);
        Icrop = imcrop(I,tshirt_box);
        if ~isempty(find(size(Icrop) > 800))
```

```
            maxim = max(size(Icrop));
            ratio = 1/(maxim/800);
            Icrop = imresize(Icrop,ratio);
        end
        for k = [1,2,3,4,5,6,7] %1: 2 %length(clubs)
            club_name = clubs(k);
            filename = sprintf('histogramas_modelo_%s.mat',club_name);
            load(filename);% loads histModels
            [existsaClubPlayer,extra_data] = isThereAClubPlayerRGB(Icrop,histModels,16,club_nar
            isClubDesition(k) = existsaClubPlayer;
            aux = [];
            aux = extra_data.good_sample;
            additional_data.good_sample = [additional_data.good_sample; aux];
            aux = [];
            aux = extra_data.bad_sample;
            additional_data.bad_sample  = [additional_data.bad_sample; aux];
        end
        n_predicted = sum(isClubDesition);
        if n_predicted == 0 || n_predicted > 3
            % it is not a good clasification when there are more than one
            % predictions
            predictedClubName = "none";
        elseif n_predicted == 3
            if isequal(isClubDesition,[1,0,0,0,1,0,1])
                predictedClubName = clubs(7); %psv
            else
                predictedClubName = "none";
            end

        elseif n_predicted == 2

            if isClubDesition(2) %  if one of the guest is barcelona , we choose to belive that
                %Barcelona
                predictedClubName = clubs(2);% Barcelona
            elseif isClubDesition(5) %liverpool
                mask = xor(isClubDesition,[0,0,0,0,1,0,0]);
                predictedClubName = clubs(find(mask));
            elseif isequal(isClubDesition,[1,0,0,0,0,0,1])
                predictedClubName = clubs(7); %psv
            elseif isequal(isClubDesition,[0,0,0,1,0,1,0])
                predictedClubName = clubs(4); %juventus
            else
                predictedClubName = "none"; % twoclubs
            end

        else
            predictedClubName = clubs(find(isClubDesition));
        end
        additional_data.exists = isClubDesition;
end

function [existsaClubPlayer, additional_data] =  isThereAClubPlayerRGB(I,histModels,bins,namecl
    [height,width,~] = size(I);
    reference_size = min(width,height);
```

```
existsaClubPlayer = false;
min_size_window = 10;
clubs = ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid", "psv"];
% [min_similar_hist,min_pct_good,min_windows_above_the_pct,min_windows_count,min_windows_co
ini_club(1,:) = [0.2,8,50,1,20,3]; %%acmilan
ini_club(2,:) = [0.1,4,45,1,7,2]; %%Barcelona
ini_club(3,:) = [0.1,4,45,1,10,2]; %Chelsea
ini_club(4,:) = [0.2,6,80,3,40,4]; %%Juventus3
ini_club(5,:) = [0.1,3,45,1,7,2]; % Liverpool
ini_club(6,:) = [0.1,14,50,2,25,4];% Madrid
ini_club(7,:) = [0.2,4,45,1,7,2];%% Psv
index = find(strcmp(clubs,nameclub));
umbrals = ini_club(index,:);
ratio = 0.9:-0.1:umbrals(1);
n_good_sample = zeros(1,9);
n_bad_sample  = zeros(1,9);
size_matrix =[];
if nameclub == "liverpool"
     I = imlocalbrighten(I);
end
for k = 1:(length(ratio)-1)
   % Test with diferent window sizes
    window_size = floor(reference_size* ratio(k));
    if window_size >= min_size_window
        shift_ratio = 1/4;
        n_col = 1/shift_ratio* floor(width/window_size) - (1/shift_ratio - 1);
        n_row = 1/shift_ratio* floor(height/window_size) - (1/shift_ratio -1);
        for i = 1:n_row
            for j = 1:n_col
                % Getting the squared window
                ini_x = (j-1)*floor(window_size*shift_ratio);
                ini_y = (i-1)*floor(window_size*shift_ratio);
                window_crop = [ini_x ini_y  window_size window_size];
                ITest = imcrop(I,window_crop);
                %threshold to set pixels that have values near 0 in 3 components to be 0
                if  index == 1 || index == 4 % Only when it is acmilan or juventus
                    mask = ITest < 50;
                    mask_black = mask(:,:,1) & mask(:,:,2) & mask(:,:,3);
                    R = ITest(:,:,1);
                    G = ITest(:,:,2);
                    B = ITest(:,:,3);
                    R(mask_black) = 0;
                    G(mask_black) = 0;
                    B(mask_black) = 0;
                    ITest(:,:,1) = R;
                    ITest(:,:,2) = G;
                    ITest(:,:,3) = B;
                end
                %Getting the color histogram in RB
                histTest = getRBColorHistogram(ITest,bins);
                histTest = imgaussfilt(histTest);
                %testing
                n_test = 15;
                % Compare and decide if it is a good or bad sample
```

```
                    is_similar = 0;
                    for q = 1:n_test
                        histoModel = histModels(:,:,q);
                        is_similar = is_similar + uint8(histSimilar(histoModel,histTest,bins,na
                    end
                    if is_similar >= umbrals(2)
                       %At leat there is histograms
                       n_good_sample(k) = n_good_sample(k) + 1 ;
                    else
                       n_bad_sample(k) = n_bad_sample(k) + 1 ;
                    end
                end
            end
        end
    end
    % Decide considereing the number of good and bad samples if there is a Barcelona's player o
    % the image
    if sum((n_good_sample./(n_good_sample + n_bad_sample))*100 >= umbrals(3)) >= umbrals(4) ||
        existsaClubPlayer = true;
    end
    additional_data.good_sample = n_good_sample;
    additional_data.bad_sample = n_bad_sample;
end


function is_similar = histSimilar(histoModel,histTest,bins,clubName)
    clubs           =   ["acmilan", "barcelona", "chelsea", "juventus", "liverpool", "madrid",
    intersec_pct    =   [0.41,      0.51,       0.2,        0.8,       0.4,        0.8,
    eucl_threshold  =   [0.04,      0.04,       0.04,       0.05,       0.08,       0.02,
    chi_threshold   =   [0.8,       0.7,        0.7,        0.7,        0.8,        0.3,

    index = find(strcmp(clubs,clubName));
    is_similar = false;
    intersection = min(histTest,histoModel);
    res_intersection = sum(intersection(:))/sum(histoModel(:));
    res_dst_eucl = dist_eucl_histograms(histoModel, histTest, bins);
    res_dst_chi = dist_chisquare_histograms(histoModel,histTest);
    is_similar = is_similar | ((uint8(res_intersection > intersec_pct(index)) + uint8(res_dst_e
end

function rgbNormalized = NormalizeRGB(I)
    I = double(I);
    max = I(:,:,1) + I(:,:,2) + I(:,:,3); %R+G+B
    max(max == 0) = 1; % preventing the problem of divide by 0 to happen
    rgbNormalized = I ./ max;
    % reparing once the problem has happend
    %rgbNormalized(rgbNormalized == Inf || rgbNormalized == NaN) = 0;
end

function dist_eucl = dist_eucl_histograms(h_A,h_B, n)
    dist_eucl = sqrt(sum(sum((h_A - h_B).^2)/n));
end

function dist_chisquare = dist_chisquare_histograms(h_A,h_B)
```

```
    % We only work with the ones that the sum is diferent than 0.
    diff_zero = (h_A + h_B) > 0;
    h_A = h_A(diff_zero);
    h_B = h_B(diff_zero);
    dist_chisquare = sum(sum( ((h_A - h_B).^2)./(h_A + h_B)));
end

function hist = getRBColorHistogram(I, bins)
    % I is an image in RGB
    aux = NormalizeRGB(I);
    I = [];
    I = aux(:,:,[1,3]);
    %range of I from  0 to 1
    [f, c, ~] = size(I);
    area = f*c;
    hist = zeros(bins, bins);
    for i=1:f
        for j=1:c
            %Classifying to the corresponding bin:
            pos_bin_red = min(floor( I(i,j,1) * bins) + 1,bins);
            pos_bin_blue = min(floor( I(i,j,2) * bins) + 1,bins);
            % Updating the bin of position bin_red and bin_blue
            hist(pos_bin_red,pos_bin_blue) = hist(pos_bin_red,pos_bin_blue) + 1;
        end
    end
    %Normalize the histogram
    if  area ~= 0
        hist = hist ./ area;
    end
end

function res = bbClub(I)
    teamData = [{0, "acmilan", bbAcmilan(I, false)}
          {0, "barcelona", bbBarca(I, false)}
          {0, "chelsea", bbChelsea(I, false)}
          {0, "juventus", bbJuventus(I, false)}
          {0, "liverpool", bbLiverpool(I, false)}
          {0, "madrid", bbMadrid(I, false)}
          {0, "psv", bbPsv(I, false)}];

    for i=1:7
        bb = teamData{i,3};
%         paintBB(I, bb, teamData(i, 2));
        teamData{i, 1} = calcRatio(bb, I);
    end
    res = selection(teamData);
end

function res = selection(teamData)

    [~, idx] = sort([teamData{:,1}], 'descend');
    sortedTeamData = teamData(idx,:);
    numElems = 1;
```

```
    if (sortedTeamData{1, 1} == 0)
        res = [];
    else
        for i=2:7
            if (sortedTeamData{i, 1} > 0.5*sortedTeamData{1, 1})
                numElems = numElems + 1;
            end
        end
        res = sortedTeamData(1:numElems, :);
        for i = 1:numElems
            res{i, 3} = largestBB(res{i, 3});
        end
    end
end

function res = largestBB(bbs)
    res = [-1, -1, -1, -1];
    max = 0;
    for k = 1 : length(bbs)
        bb = bbs(k).BoundingBox;
        area = bb(3)*bb(4);
        if (area > max)
            max = area;
            res = bb;
        end
    end
end

function res = calcRatio(bbs, I)
    mida = size(I);
    numPixels = mida(1)*mida(2);
    area = 0;
    for k = 1 : length(bbs)
        bb = bbs(k).BoundingBox;
        area = area + bb(3)*bb(4);
    end
    res = min(area/numPixels, 1);
end

function res = bbAcmilan(I, debug)
    mida = size(I);

%     Inorm = NormalizeRGB(I)*255;

%     IvermellNormBin = rgb2binByColor(Inorm, [199, 26, 30]);
    IvermellBin = rgb2binByColor(I, [173, 22, 27]);

    if (debug)
        figure;
        imshow(IvermellBin); title("Binarizacion segun el color rojo del Milan");
    end

%     InegreNormBin = rgb2binByColor(Inorm, [75, 91, 89]);
    InegreBin = rgb2binByColor(I, [15, 15, 15]);
```

```
    if (debug)
        figure;
        imshow(InegreBin); title("Binarizacion segun el color negro del Milan");
    end

    IBin = binTotal(IvermellBin, InegreBin);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion resultado de la operacion magica");
    end


    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.5;
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " Milan");
    end

    numPixels = numel(IBin);
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles Milan");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = bbBarca(I, debug)
    mida = size(I);
    Inorm = NormalizeRGB(I)*255;

    IvermellBin = rgb2binByColor(Inorm, [145, 50, 60]);

    if (debug)
        figure;
        imshow(IvermellBin); title("Binarizacion segun el color rojo del Barça");
    end

    IblauBin = rgb2binByColor(Inorm, [44, 71, 140]);

    if (debug)
```

```
        figure;
        imshow(IblauBin); title("Binarizacion segun el color azul del Barça");
    end

    IBin = binTotal(IvermellBin, IblauBin);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion resultado de la operacion magica");
    end

    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.5;
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " Barça");
    end

    numPixels = numel(IBin);
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles Barça");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = bbChelsea(I, debug)
    mida = size(I);

    Inorm = NormalizeRGB(I)*255;

    %IBin = rgb2binByColor(I, [85, 99, 208]);
    IBin = rgb2binByColor(Inorm, [55, 64, 135]);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion segun el color azul del Chelsea");
    end


    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.3;
```

```
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " Chelsea");
    end

    numPixels = numel(IBin);
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles Chelsea");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = bbJuventus(I, debug)
    mida = size(I);

    IblancBin = rgb2binByColor(I, [240, 240, 240]);

    if (debug)
        figure;
        imshow(IblancBin); title("Binarizacion segun el color blanco del Juve");
    end

    InegreBin = rgb2binByColor(I, [40, 40, 40]);

    if (debug)
        figure;
        imshow(InegreBin); title("Binarizacion segun el color negro del Juve");
    end

    IBin = binTotal(IblancBin, InegreBin);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion resultado de la operacion magica");
    end


    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.5;
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
```

```
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " Juve");
    end

    numPixels = numel(IBin);
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles Juve");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = bbLiverpool(I, debug)
    mida = size(I);

    Inorm = NormalizeRGB(I)*255;

    %IBin = rgb2binByColor(I, [172, 24, 35]);
    IBin = rgb2binByColor(Inorm, [190, 26, 39]);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion segun el color rojo del Liverpool");
    end


    if (debug)
        figure;
        imshow(IBin); title("Binarizacion resultado de la operacion magica");
    end


    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.8;
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " Liverpool");
    end

    numPixels = numel(IBin);
```

```
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles Liverpool");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = bbMadrid(I, debug)
    mida = size(I);

    IBin = rgb2binByColor(I, [240, 240, 240]);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion segun el color blanco del Madrid");
    end


    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.3;
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " Madrid");
    end

    numPixels = numel(IBin);
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles Madrid");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = bbPsv(I, debug)
    mida = size(I);
    Inorm = NormalizeRGB(I)*255;

    IblancBin = rgb2binByColor(I, [240, 240, 240]);
```

```
%     IblancNormBin = rgb2binByColor(Inorm, [85, 85, 85]);

    if (debug)
        figure;
        imshow(IblancBin); title("Binarizacion segun el color blanco del PSV");
    end

%     IvermellBin = rgb2binByColor(I, [160, 25, 25]);
    IvermellBin = rgb2binByColor(Inorm, [175, 39, 41]);

    if (debug)
        figure;
        imshow(IvermellBin); title("Binarizacion segun el color rojo del PSV");
    end

    IBin = binTotal(IblancBin, IvermellBin);

    if (debug)
        figure;
        imshow(IBin); title("Binarizacion resultado de la operacion magica");
    end


    % indica el porcentaje del radio del disco respecto al lado mínimo de la
    % imagen
    porcentajeDisco = 0.5;
    radioDisco = round((min(mida(1), mida(2))*porcentajeDisco)/100);

    SE = strel('disk', radioDisco);
    IBin = imclose(IBin, SE);

    if (debug)
        figure;
        imshow(IBin); title("IMCLOSE con radio "+num2str(radioDisco)+ " PSV");
    end

    numPixels = numel(IBin);
    porcentajeDeAreaMinima = 1;
    numPixelsMinimo = round((numPixels*porcentajeDeAreaMinima)/100);
    IBin = bwareaopen(IBin, numPixelsMinimo);

    if (debug)
        figure;
        imshow(IBin); title("Filtro de "+num2str(numPixelsMinimo)+" pixeles PSV");
    end

    res = regionprops(IBin, 'BoundingBox');
end

function res = rgb2binByColor(I, color)
    threshold = 40;
    I = double(I);
    mida = size(I);
    res = false(mida(1), mida(2));
```

```
    for i = 1:mida(1)
        for j = 1:mida(2)
            val = [I(i, j, 1), I(i, j, 2), I(i, j, 3)];
            dist = euDistColor(val, color);
            res(i, j) = dist < threshold;
        end
    end
end

function res = binTotal(binVermell, binBlau)
    mida = size(binVermell);
    res = false(mida(1), mida(2));
    for i = 1:mida(1)
        for j = 1:mida(2)
            if (binVermell(i, j) || binBlau(i, j))
                res(i, j) = sufficientWhite(i, j, binVermell, binBlau);
            end
        end
    end
end

function res = sufficientWhite(iOri, jOri, vermell, blau)
    mida = size(vermell);

    percentatgeThreshold = 0.4;

    % El numero de pixeles minimo que tiene que contar es proporcional al
    % numero de pixeles de la imagen
    threshold = (mida(1)*mida(2)*percentatgeThreshold)/100;
    percentatgeMidaMin = 10; % Determina el tamaño del area a tener en cuenta

    lengthWidth = uint32(mida(2)*(percentatgeMidaMin/100));
    lengthHeight = uint32(mida(1)*(percentatgeMidaMin/100));
    numBlaus = 0;
    numVermells = 0;
    for i = max(1, iOri-lengthHeight):min(mida(1), iOri+lengthHeight)
        for j = max(1, jOri-lengthWidth):min(mida(2), jOri+lengthWidth)
            if (vermell(i, j))
                numVermells = numVermells+1;
            elseif (blau(i, j))
                numBlaus = numBlaus+1;
            end
        end
    end
    res = numBlaus >= threshold && numVermells >= threshold;
end

function res = euDistColor(c1, c2)
    res = sqrt((c1(1)-c2(1))^2 + (c1(2)-c2(2))^2 + (c1(3)-c2(3))^2);
end
```