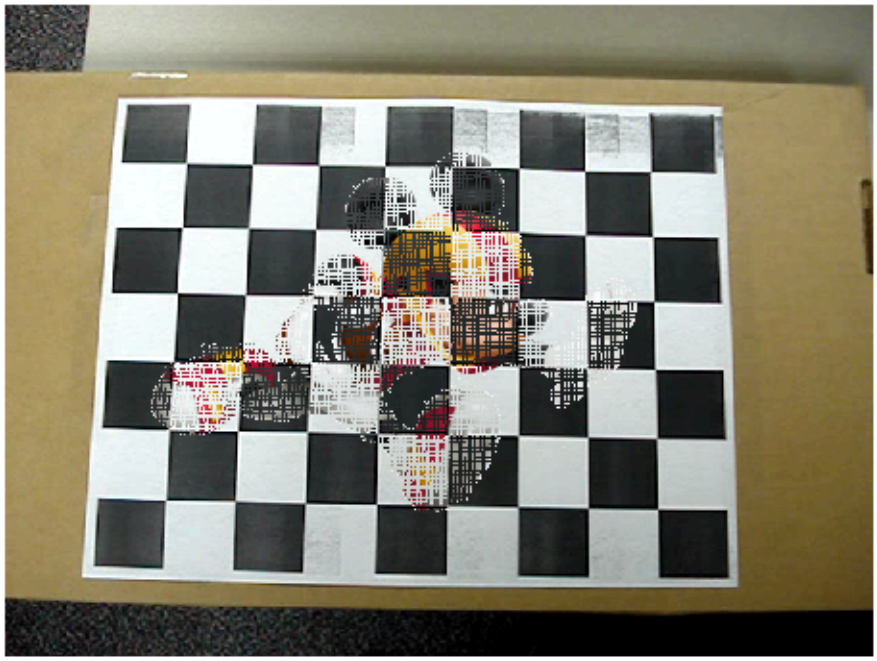
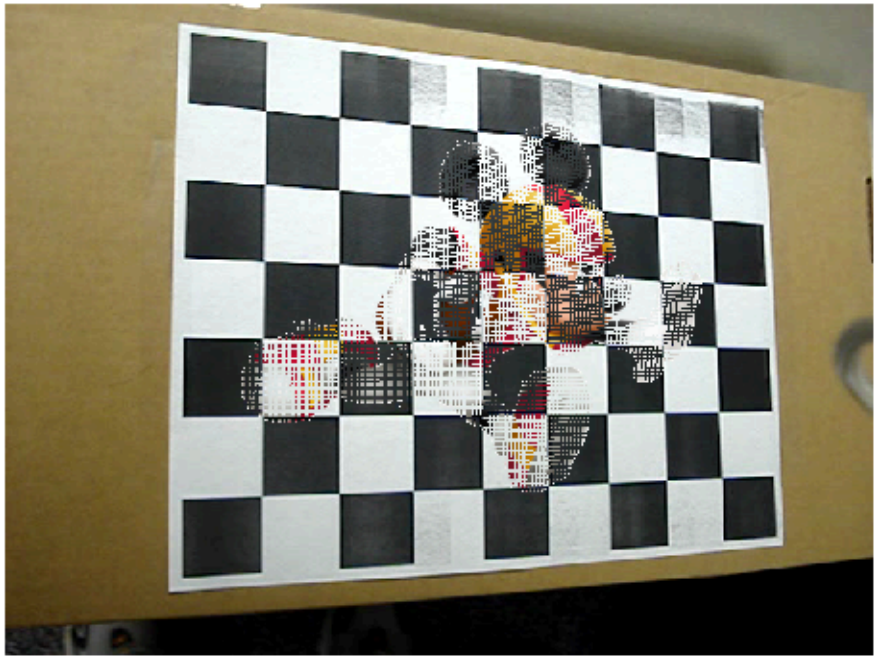
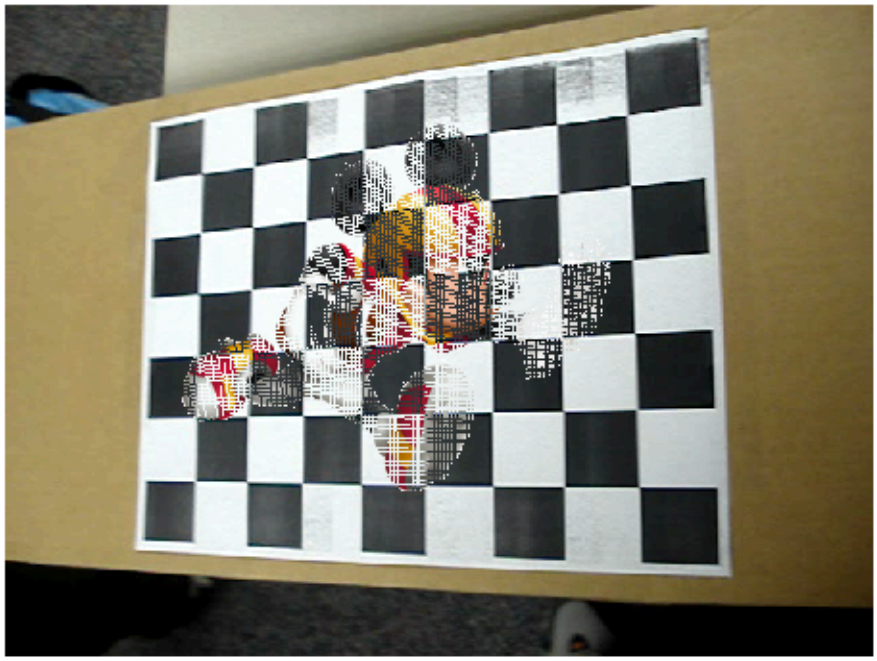
```
%Read the assigned clipart
cart = imread("3.jpg");
%Scale the clipart to match the height of the chessboard pattern
scale = [height*30 NaN];
%Resize the clipart to the height of the grid without modifying the
 width
%hence ensuring that we maintain the clip art aspect ratio
cart = imresize(cart,scale);
%Obtain the number of rows and columns of the clipart
[rowsM, colsM, channels_clipart] = size(cart);

cart = cart + 1;
%Change from [y,x] point notation to [x,y] notation.
cart = imrotate(cart,-90);

for k=1:4
    %Get the corresponding read image
    img = eval(['img' num2str(k)]);
    %Get size -- rows and columns of the image
    [rowsG, colsG, channels_img] = size(img);
    %Get the recomputed homographies from the Harris Corner detection
    H = eval(['Hnew' num2str(k)]);
    cart_points = double([]);

    for i = 1:colsM
        for j = 1:rowsM
            %Obtain all the corresponding pixels points of the clipart
 with
            %its RGB values.
            cart_points = [cart_points;i, j, 1, double(cart(i,j,1)),
 double(cart(i,j,2)), double(cart(i,j,3))];
        end
    end

    %Get the projected points for the clipart by multiplying the
 [X,Y,1]
    % with the homography H
    clipart_projected = double(cart_points(:,1:3))*H';
    %Normalize the X & Y projections by doing a right array divison
    clipart_projected(:,1) = clipart_projected(:,1)./
clipart_projected(:,3);
    clipart_projected(:,2) = clipart_projected(:,2)./
clipart_projected(:,3);
    %Find the absoluted closest integer value of the projected points
    clipart_projected = round(clipart_projected+1);

    %Create an augment layer matrix having the number of rows and
 columns
    %as required after obtaining the projected points
    augmented_layer = zeros(max(clipart_projected(:,1)),
 max(clipart_projected(:,2)), 3);
```

```
    %Run a loop for all the projected pixel points.
    for i = 1:length(clipart_projected)
        %Get the row and column pixel projection onto the image
        r = clipart_projected(i,2);
        c = clipart_projected(i,1);
        %Set the corresponding channels of the projected pixel points
 to
        %have the RGB values of the clipart
        augmented_layer(r,c,1) = cart_points(i,4);
        augmented_layer(r,c,2) = cart_points(i,5);
        augmented_layer(r,c,3) = cart_points(i,6);
    end

    %Conver the augmented layer from double to uint8
    augmented_layer = uint8(augmented_layer);
    value = augmented_layer;
    %Considers only the non-white pixels
    bw = (value > 0) & (value < 255);
    %Gets the difference between the original image and our clipart
 with
    %white pixels removed
    diff = abs(size(img)-size(value));
    %Creates a padded array with zeros where there are no clipart
 pixels to
    %where the extra background white pixels where removed
    % "post" - Pad after the last array element along each dimension.
    mask_pad = padarray(bw, [diff(1),diff(2)], "post");
    img_masked_pad = value .* uint8(bw);

    %Create a new image matrix of the size of the original image
    new_img = zeros(rowsG,colsG,3);
    %Run a loop for all the columns
    for i = 1:rowsG
    for j = 1:colsG
        %If value of mas_pad is anything other than a zero
        %then, copy the clipart color values to the new image
        if mask_pad(i,j,:)
            new_img(i,j,:) = img_masked_pad(i,j,:);
        %Else, ccopy the original image color values
        else
            new_img(i,j,:) = img(i,j,:);
        end
    end
    end
    %Display the new image generated after the projection of the
 clipart
    figure(), imshow(uint8(new_img))
end
```







Published with MATLAB® R2019a