Laboratory of Image Processing

Video Mosaicking

Pier Luigi Mazzeo

pierluigi.mazzeo@cnr.it

Institute of Applied Sciences and Intelligent Systems

ScienceApp

National Research Council of Italy

Mosaicking

- This example shows how to create a mosaic from a video sequence. Video mosaicking is the process of stitching video frames together to form a comprehensive view of the scene. The resulting mosaic image is a compact representation of the video data, which is often used in video compression and surveillance applications.

- This example illustrates how to use **detectSURFFeatures**, **extractFeatures**, **matchFeatures**, and **estimateGeometricTransform** to create a mosaic image from a video sequence. First, the example identifies the corners in the first (reference) and second video frames. Then, it calculates the affine transformation matrix that best describes the transformation between corner positions in these frames. Finally, the example overlays the second image onto the first image. The example repeats this process to create a mosaic image of the video scene.

Inizialization

```
%% Initialization
% Define the size and location of the output mosaic
image.
[w, h]      = deal(680, 400);  % Size of the mosaic
[x0, y0]    = deal(-5, -60);   % Upper-left corner of the
mosaic
xLim = [0.5, w+0.5] + x0;
yLim = [0.5, h+0.5] + y0;
outputView = imref2d([h,w], xLim, yLim);
```

Inizialization

```
%%
% Create a VideoFileReader System object to
read video from a file.
hsrc =
vision.VideoFileReader('vipmosaicking.avi',
'ImageColorSpace', ...
    'RGB', 'PlayCount', 1);

%%
% Create a AlphaBlender System object to
overlay the consecutive video
% frames to produce the mosaic.
halphablender = vision.AlphaBlender( ...
    'Operation', 'Binary mask', 'MaskSource',
'Input port');
```

Inizialization

```
%%
% Create two VideoPlayer System objects, one to display the
corners of each
% frame and other to draw the mosaic.
hVideo1 = vision.VideoPlayer('Name', 'Corners');
hVideo1.Position(1) = hVideo1.Position(1) - 350;

hVideo2 = vision.VideoPlayer('Name', 'Mosaic');
hVideo2.Position(1) = hVideo1.Position(1) + 400;
hVideo2.Position([3 4]) = [750 500];

%%
% Initialize some variables which will be used later.
points = cornerPoints(zeros(0, 2));
features = single(zeros([0 64]);
failedToMatchPoints = true; % A new mosaic will be created if%
failedToMatchPoints is true
```

Stream Processing Loop

```
% Create a processing loop to create mosaic from the input
video. This
% loop uses the System objects you instantiated above.
while ~isDone(hsrc)
    % Save the points and features computed from the previous
image
    pointsPrev   = points;
    featuresPrev = features;

    % To speed up mosaicking, select and process every 5th
image
    for i = 1:5
        rgb = step(hsrc);
        if isDone(hsrc)
            break;
        end
    end

    % Convert the image from RGB to intensity.
    I = rgb2gray(rgb);
```

Stream Processing Loop

```
% Detect corners in the image
    corners = detectSURFFeatures(I);

    % Extract SURF feature vectors for the corners
    [features, points] = extractFeatures(I, corners);

    % Match features computed from the current and the
previous images
    indexPairs = matchFeatures(features, featuresPrev);

    % Check if there are enough corresponding points in the
current and the
    % previous images
    if size(indexPairs, 1) > 2
        matchedPoints     = points(indexPairs(:, 1), :);
        matchedPointsPrev = pointsPrev(indexPairs(:, 2), :);
```

Stream Processing Loop

```
 % Find corresponding locations in the current and the
previous images, and compute a geometric transformation from
the corresponding locations
        [tform, ~, ~, failedToMatchPoints] =
estimateGeometricTransform(...
            matchedPoints, matchedPointsPrev, 'affine');
    end

 if failedToMatchPoints
% If the current image does not match the previous one, reset
the transformation and the mosaic image
        xtform = eye(3);
        mosaic = zeros(h, w, 3, 'single');
    else
        % If the current image matches with the previous one,
compute the transformation for mapping the current image onto
the mosaic image
        xtform = xtform * tform.T;
    end
```

Display results

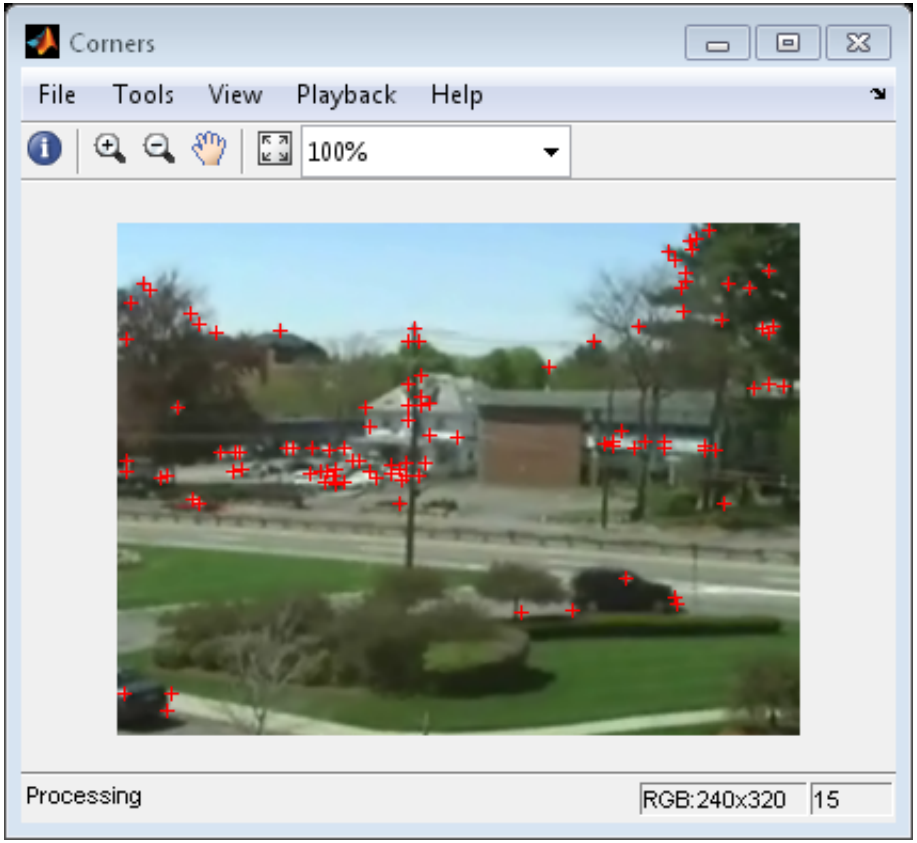
```
 % Display the current image and the corner points
    cornerImage = insertMarker(rgb, corners.Location,
'Color', 'red');
    step(hVideo1, cornerImage);

    % Creat a mask which specifies the region of the
transformed image.
    mask = imwarp(ones(size(I)), affine2d(xtform),
'OutputView', outputView) >= 1;

    % Warp the current image onto the mosaic image
    transformedImage = imwarp(rgb, affine2d(xtform),
'OutputView', outputView);
    mosaic = step(halphablender, mosaic,
transformedImage, mask);
    step(hVideo2, mosaic);
end

release(hsrc);
```

The Obtained Results



The Obtained Results