# Feature Tracker

Aim of this project was to track features in a series of image frames. MatLab software tool was used to implement the algorithm. The processing approach used in this project have two main tasks.

1. Detect the features from the first frame.
2. Follow the detected features from frame to frame in the series of image frames.

## Detecting Features

Harris corner detection method was selected to detect features for tracking. Because it is the most popular method for detect features.

Idea Harris detector – change of intensity for the shift [u, v]:



Simplify using tailor expansion:





Corner response Function:

Here “R” is the corner response function of the image while M is the second moment matrix of the image, λ1 & λ2 are the eigenvalues of M. “k” is a constant value and here in this project its value was 0.06.

%%Find Ix and Iy with smoothing

dx =[-1 0 1 ; -1 0 1 ; -1 0 1];

dy =[-1 -1 -1 ; 0 0 0 ; 1 1 1 ];

Ix = imfilter(im,dx);

Iy = imfilter(im,dy);

%%finding gaussian filtered Ixx Iyy and Ixy

Ixx = imfilter(Ix.\*Ix,gaussian);

Iyy = imfilter(Iy.\*Iy,gaussian);

Ixy = imfilter(Ix.\*Iy,gaussian);

%Corner response function

R= (Ixx.\*Iyy - Ixy.^2) - k\*(Ixx + Iyy).^2;

After obtaining the corner response function, local non-maximum suppression was done over a 5x5 window centered at each point. Thresholding also applied to the image at the same time using a predefined threshold value of 0.000005. Then X,Y coordinates of the feature points were saved in two separate arrays.

%compute the local maxima of R above a threshold 5-by-5 windows

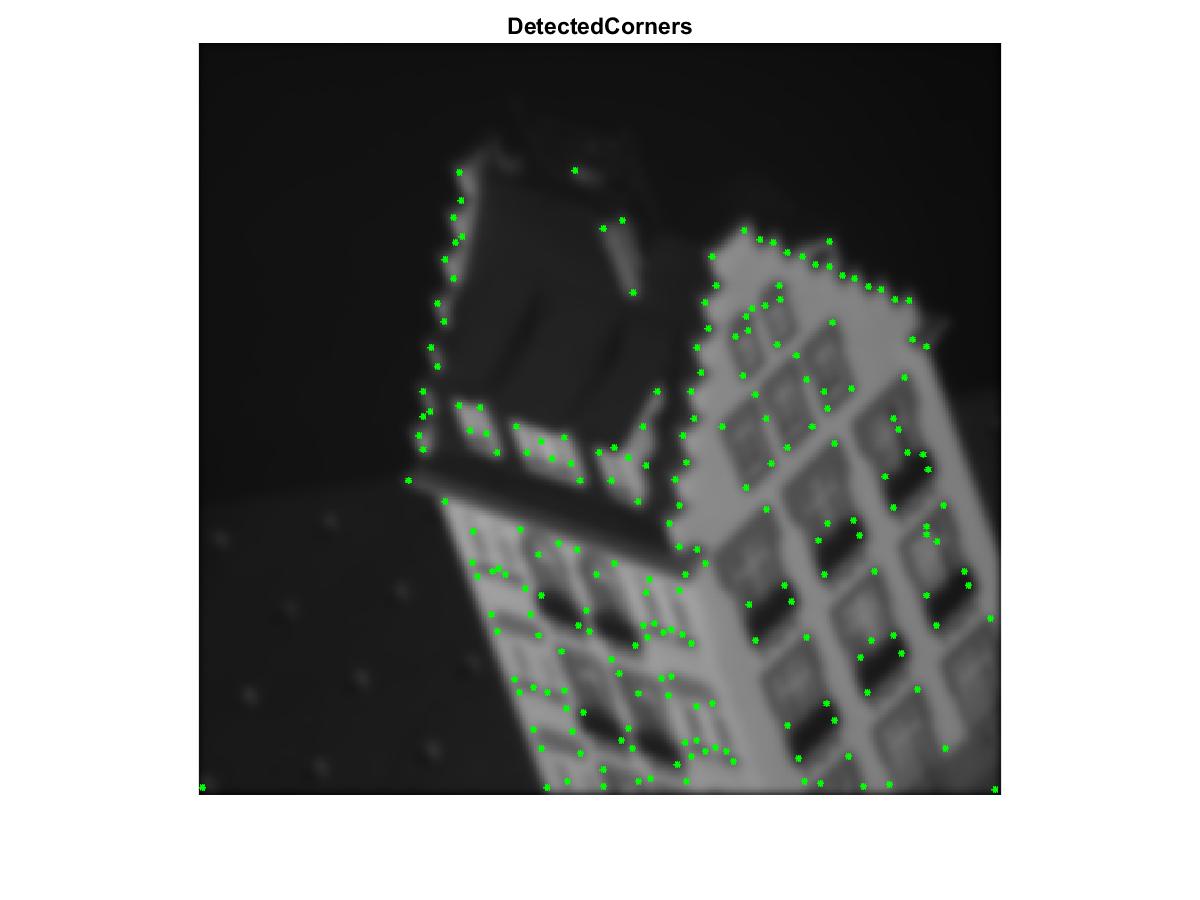
corners = ordfilt2(R, 25, ones(5));

mask = (R == corners) & (R > tau) ;

corners = mask.\*R;

[keyYs, keyXs] = find(corners>0);

**Results of Feature Detection**



## Tracking Features

1. Select features to track

Before feature tracker starts working, it selects 40 random points from the total detected key points

%%Select Random Key Points

pts = randperm(size(keyYs,1));

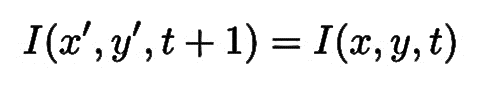
pts = pts(1:40);

keyXs\_rand = keyXs(pts);

keyYs\_rand = keyYs(pts);

1. Track features using KLT algorithm

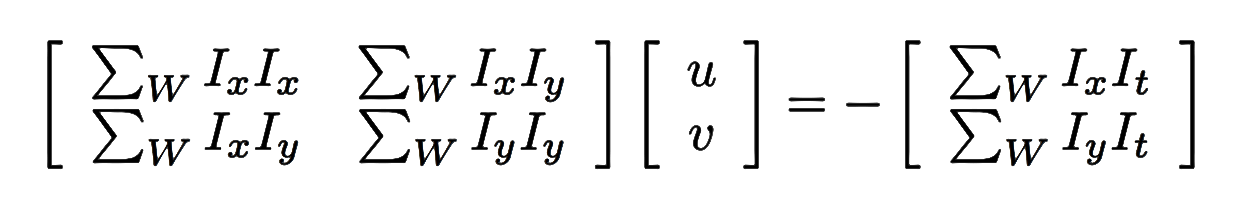
* Idea of KLT Algorithm (Kanade-Lucas-Tomasi tracking procedure) is to, compute the expected translation from: (x, y) => (x’, y’), of each key point for every frame.



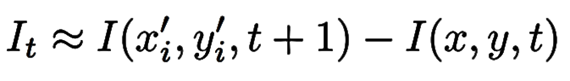
* Key assumptions of Kanade-Lucas-Tomasi Tracker
  1. Brightness constancy: projection of the same point looks the same in every frame
  2. Small motion: points do not move very far
  3. Spatial coherence: points move like their neighbors
* Brightness Constancy Equation:



* want to update our estimate (x’i+1, y’i+1) = (x’i, y’i) + (u, v) Iteratively using following equation

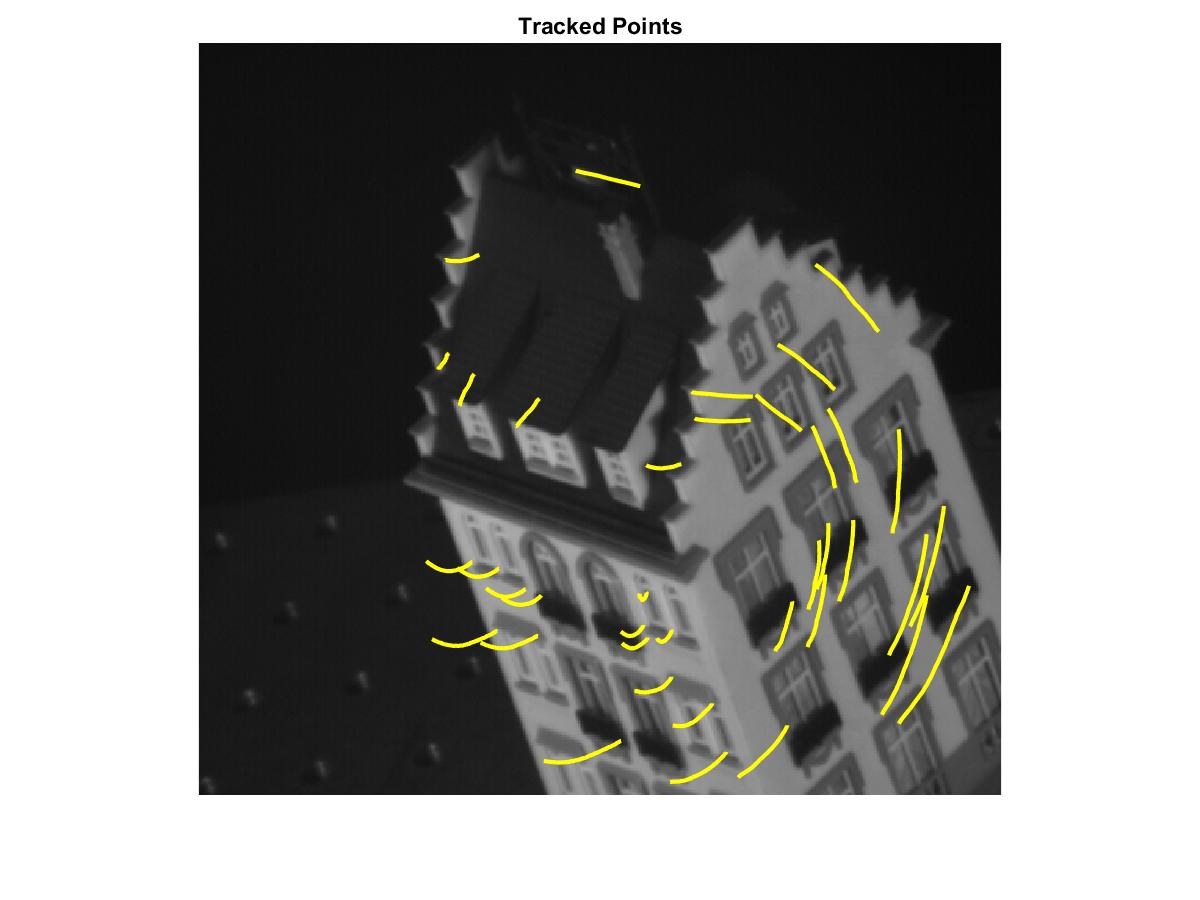


* Here It (temporal gradient),



* W is a 15x15 pixel window surrounding the key point.
* Ix, Iy are the x, y gradients of image I(x, y, t).

1. Results





## References

* <https://courses.engr.illinois.edu/cs543/sp2015/>
* <https://courses.engr.illinois.edu/cs543/sp2012/lectures/Lecture%2008%20-%20Feature%20Tracking%20and%20Optical%20Flow%20-%20Vision_Spring2012.pdf>

## GitHub Link

* <https://github.com/nisaldilshan/Feature-Tracker>