Independent motion detection

Algorithm

My algorithm first use the build in function cv2. goodFeaturesToTrack() to detect the corners and cv2.calcOpticalFlowPyrLK to implement L-K algorithm. Here, I set pyramid=2 to 4 to try which one is better. **When I find the matched optical flow, I calculate the norm of flow vectors. If over 50% percent optical flow stay at its original position (Here I make the tolerance=1), then I have confidence to make sure that this camera is stationary. And similarly, the camera moves when most of points are moving.**

If my alg. determine the camera moves, then my algorithm will use RANSAC to find all the inliers. And I use this inliers to calculate the FOE. I use the formula FOE= (A^T A)^-1 A^T b to calculate the FOE. Here A consist of previous corners and b is the flow vectors. Once I get the FOE, my alg. will use cv2.circle () to draw it on the original image. Then my algorithm will consider those outliers as independent objects or error.

When I got these motion region. I use the meanshift clustering algorithm to find the cluster centers. And each cluster centers should be the independent objects. Also, I set the threshold for each cluster centers. If less than 4 flow vectors in the cluster center, then my algorithm will delete this center automatically.

Finally, my algorithm will use different color for different clusters and use cv2.rectangle with size (40x40) to draw a bounding box.

Results

pairs: 0 1

Most motion vectors are moving

camera is moving

pairs: 2 3

Most motion vectors are moving

camera is moving

pairs: 6 7
Most motion vectors are moving
camera is moving

pairs: 4 5
Above 50% motion vectors are stay in original position
camera is stationary

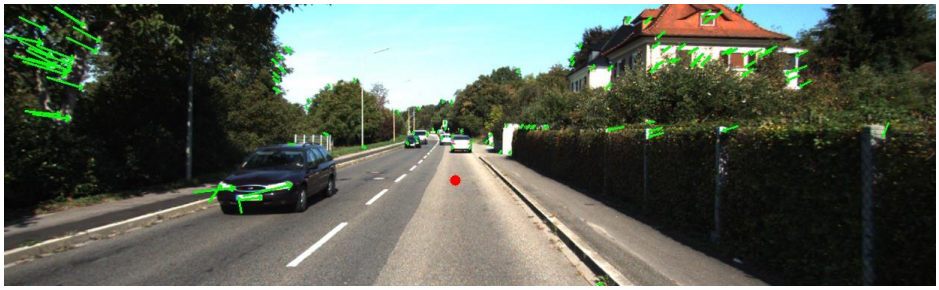
pairs: 8 9
Most motion vectors are moving
camera is moving

pairs: 10 11
Most motion vectors are moving

camera is moving

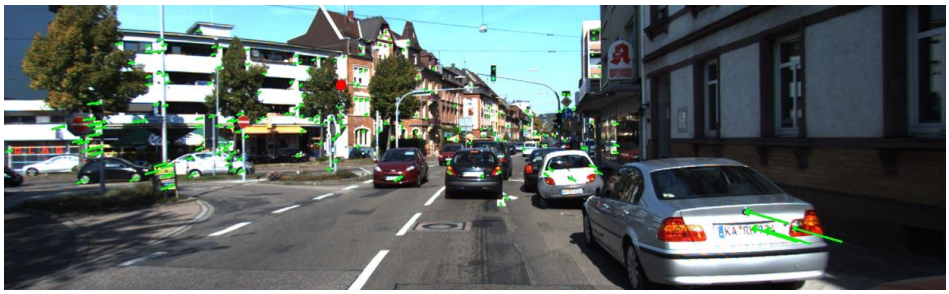
pairs: 12 13
Most motion vectors are moving
camera is moving

pairs: 14 15
Most motion vectors are moving
camera is moving



pairs: 16 17
Most motion vectors are moving
camera is moving

pairs: 18 19
Most motion vectors are moving
camera is moving



pairs: 20 21
Most motion vectors are moving
camera is moving

pairs:22 23
Most motion vectors are moving
camera is moving

pairs: 24 25
Most motion vectors are moving
camera is moving

pairs: 26 27
Above 50% motion vectors are stay in original position
camera is stationary

pairs: 28 29
Above 50% motion vectors are stay in original position
camera is stationary

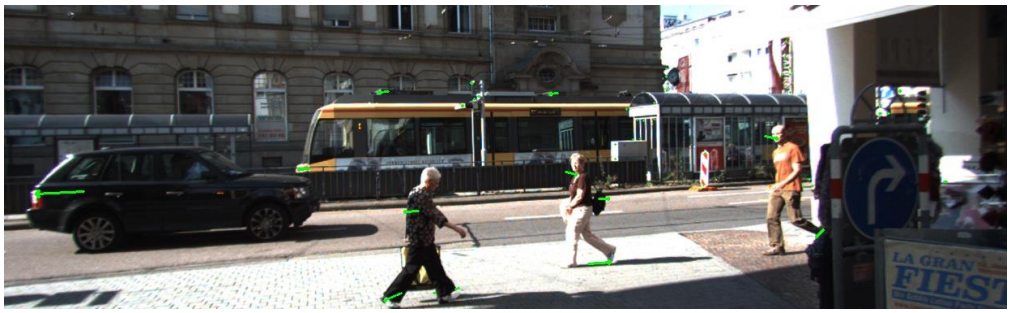
pairs: 30 31
Most motion vectors are moving
camera is moving

pairs: 32 33
Most motion vectors are moving
camera is moving

pairs: 34 35

Most motion vectors are moving

camera is moving



pairs: 36 37

Above 50% motion vectors are stay in original position

camera is stationary

pairs: 38 39

Most motion vectors are moving

camera is moving

pairs: 40 41

Most motion vectors are moving

camera is moving

pairs: 42 43

Most motion vectors are moving

camera is moving

pairs: 44 45

Most motion vectors are moving

camera is moving

pairs: 46 47

Most motion vectors are moving

camera is moving

pairs: 48 49

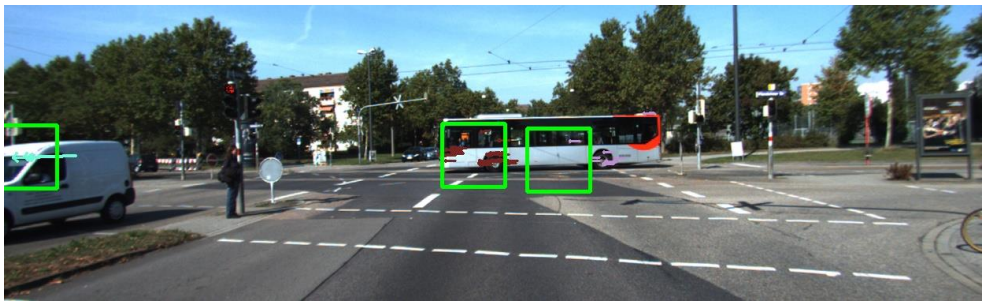
Most motion vectors are moving

camera is moving

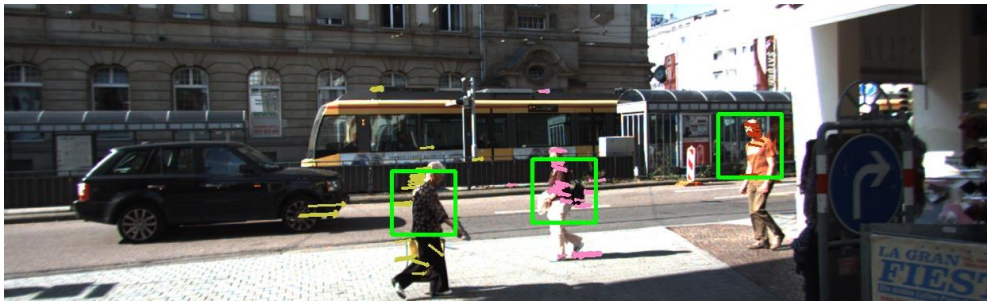
From above results, we can find that if camera is stationary, then most of flow vectors are actually a point. When I calculate the norm, they all less than 1. And the independent objects are very obvious. Because the scene do not change and only the motion of objects will produce a large

optical flow. If the camera is moving, I observe that most of flow vectors come towards the FOE and FOE is the intersection. And the vectors that close to the FOE are shorter. That is because vectors close to the center, the change become very small when compared with vectors which far from the FOE.

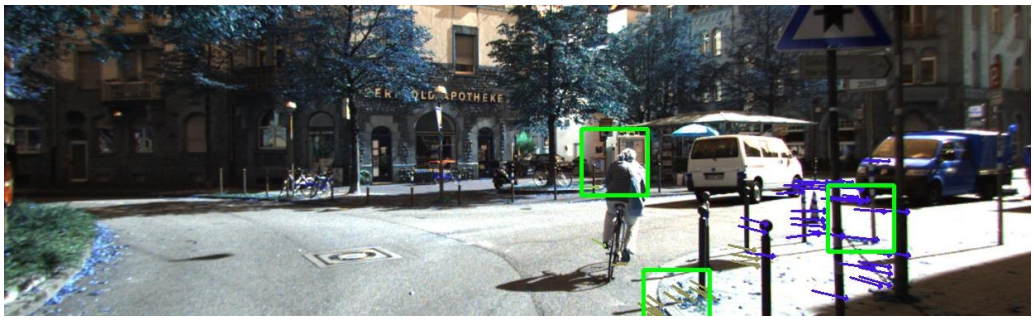
My algorithm can find all the motion objects when camera is stationary.

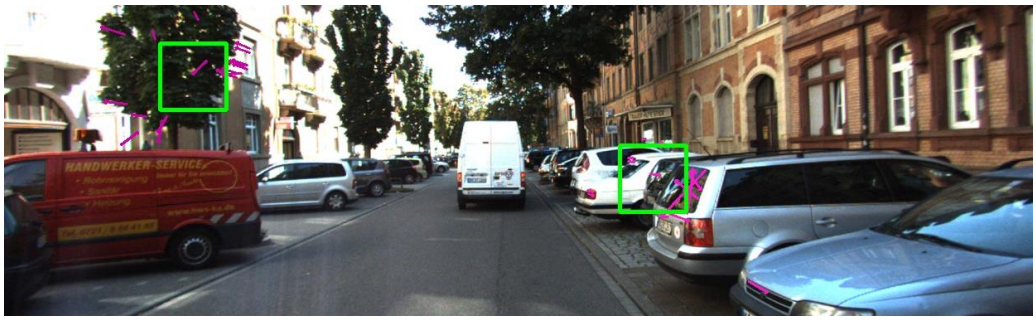


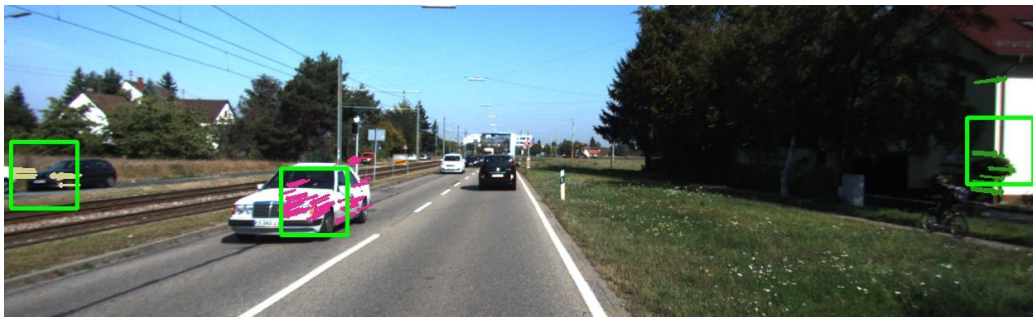
However, this image seems like miss the car and the train. Since my algorithm clustering need each cluster centers having more than 4 flow vectors. Therefore, from the image we can see that there is less than 4 vectors on the car and train. And also in image pairs 24, the human moves, but my algorithm do not detect it. I think the reason is L-K algorithm assume the brightness of image unchanged. However, the human lighter in first frame than next frame. Therefore, it might miss this motion.

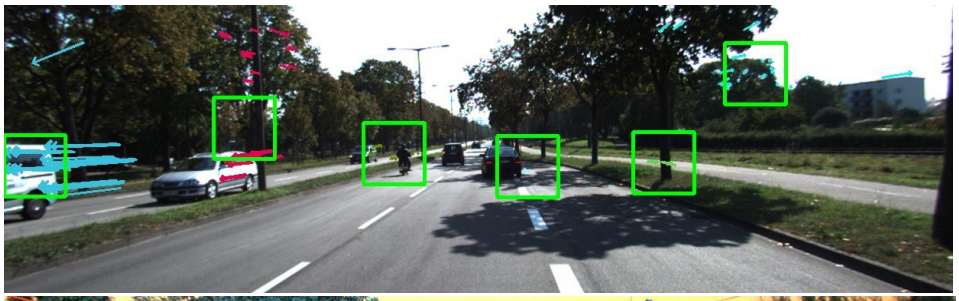


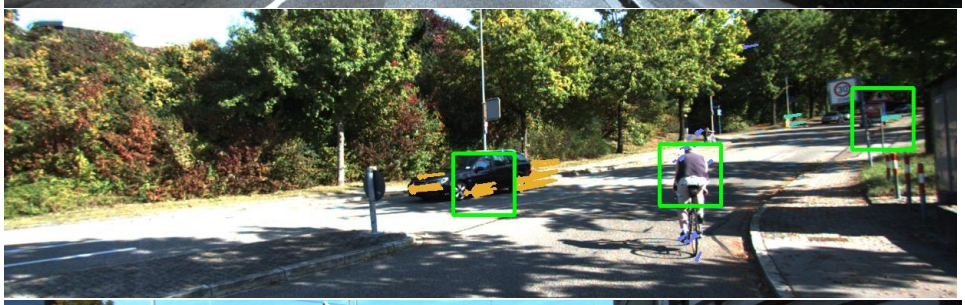
And one big challenge is the process that algorithm determine the independently objects. Although, I also use ORB descriptor to help me find the motion region. There are still existing some errors. The errors as followed.

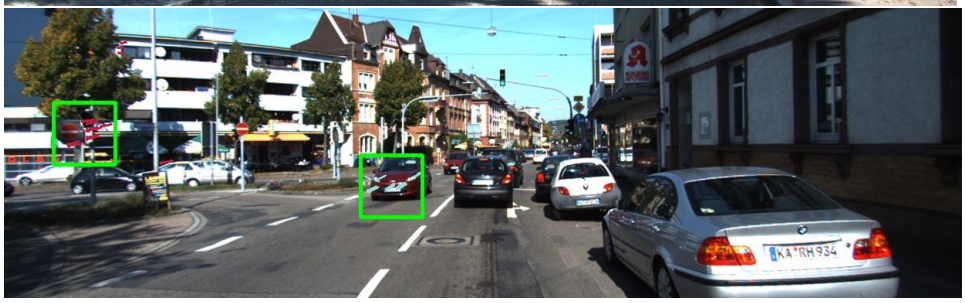


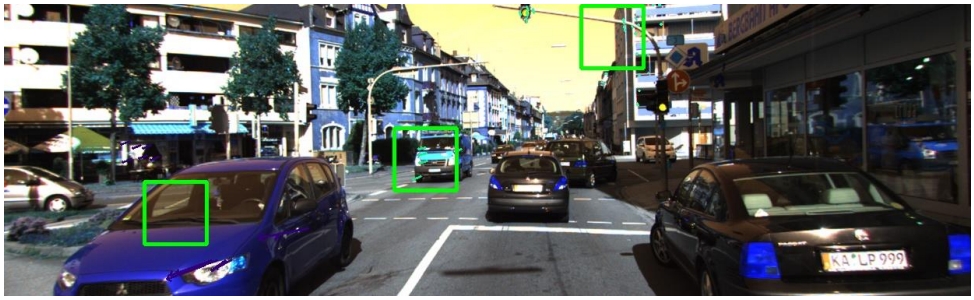


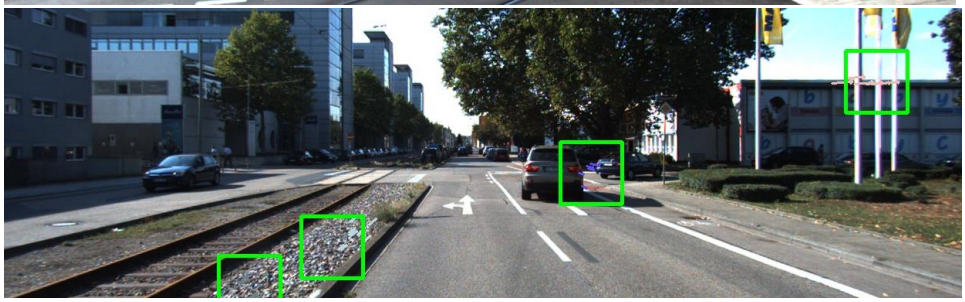


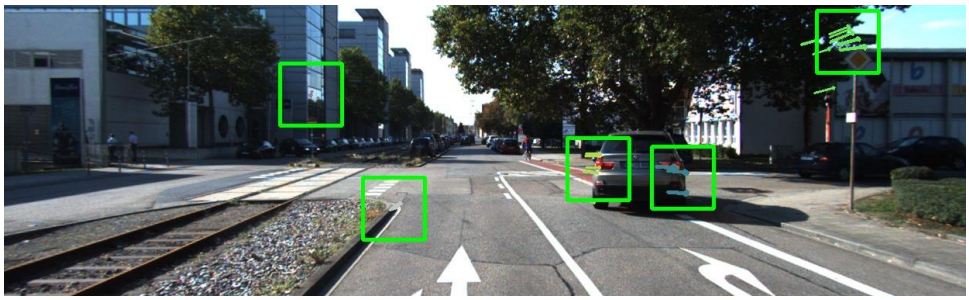


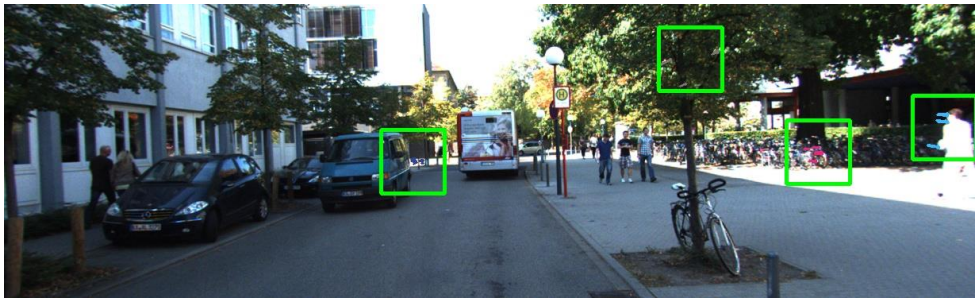


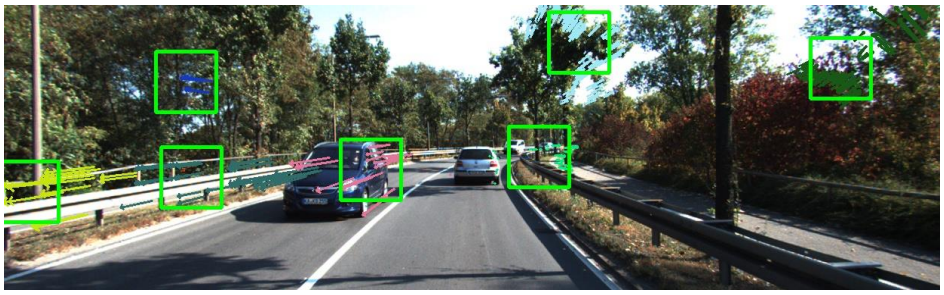


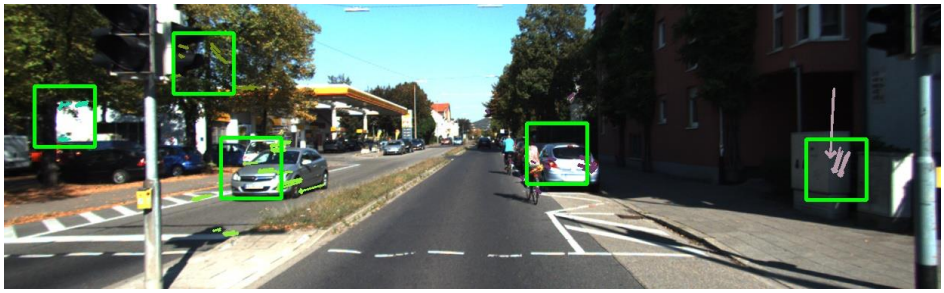












These results have a common feature that the error are always far from the FOE. And from the results we can see that the region far from the FOE still have lots of flow vectors. **The weakness of my algorithm is hard to tell the error and the independent objects. The clustering algorithm that I use is my advantage. Because only if my algorithm can find the motion region, then this algo. can easy determine the cluster centers. And it is unsupervised, which is really powerful.**

These results are pretty clean because the environment is pretty dark. And it is hard to find those corners in the dark. And the L-K algorithm will focus on the light part. Therefore, it won't produce some noises when match the optical flow.