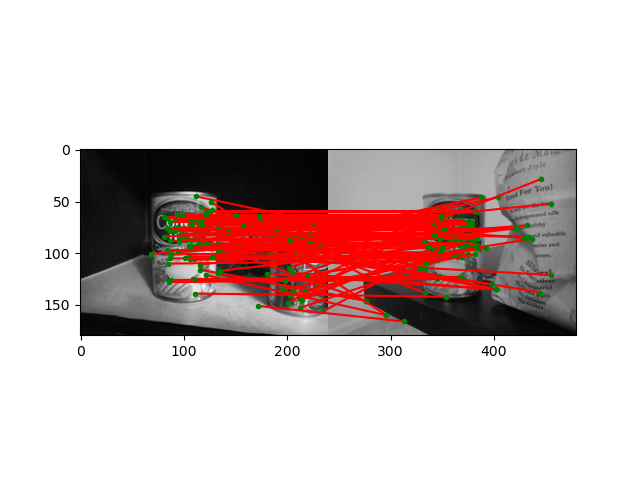
אלגוריתמים בראייה ממוחשבת

046746

רטוב 1

דניאל טייטלמן – 207734088 Daniel.tei@campus.technion.ac.il

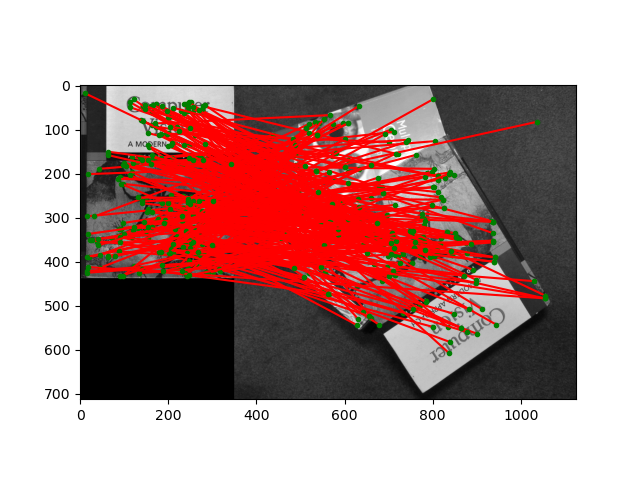
יאיר נחום – 034462796 nahum.yair@campus.technion.ac.il

1. Fdsgfd
2. Part B – Brief Descriptor (code is at code/part\_B/ the results are at data/my\_data/)
   1. See implementation in my\_BRIEF.py code. The main.py module uses it and creates the testPattern.mat file at every run of main.py   
      From the article methods we’ve implemented the uniform distribution randomization on patch locations.
   2. See in my\_BRIEF.py. We’ve implemented comupteBrief function by first filtering out any point that doesn’t have the required support (9x9) within the image pixels’ boundaries.  
      Then, we’ve built the ‘m’ locations’ descriptors:  
      We’ve converted the 9x9 support per location to a flat vector of 81x1. Then, we’ve taken the 2 series of pairs, Xn and Yn, and get the vector according the Rho function defined (to get the binary vector as a descriptor per feature location). It could have been bits (to save memory and get better performance on distance calculations) but we’re not required to save in bits.
   3. Here, we simply load the testPattern.mat in which we save the parameters for briefLite function (not only compareX and compare fields in hash).   
      We then get the DoG features and the Gaussian pyramid and use them together with other parameters to calculate the filtered (inner pixels) features’ locations and their descriptors.
   4. Descriptors’ matching  
      We’ve implemented the testMatch.py script (also run from part B main.py) and plotted the results:
      1. Chickenbroth image 01 vs image 04:  
           
         In the above example we can see that the brief is robust and invariant against illumination changes. This is so, since we calculate the R value in DoG filtering as a ratio between trace^2 and det of the gradient’s covariance matrix. Thus, as we saw in lecture, it will be invariant to scale and illumination (better in this sense from Harris Detector).
      2. incline\_L vs incline\_R:  
         A picture containing surface chart

         Description automatically generated  
         We can see that BRIEF works well also when there is a different perspective on the scene. It matches the same building.
      3. pf\_scan\_scaled vs pf\_desk:Graphical user interface, chart, surface chart

         Description automatically generated  
         under perspective transformation, it seems that the feature points are preserved and match.
      4. pf\_scan\_scaled vs pf\_floor:A picture containing diagram

         Description automatically generated  
         Since the CV book image is not rotated on the floor, we can see that the BRIEF operates very good. It also finds some false matches against other books though.
      5. pf\_scan\_scaled vs pf\_floor\_rot:  
         A picture containing diagram

         Description automatically generated  
         When rotation is applied, we see the performance degrade and BRIEF can’t match the correct book over the others. It finds matches similarly against all books.
      6. pf\_scan\_scaled vs pf\_pile:  
         Again, poor performance when rotating. Now the shape of the book against the background is also not clear and BRIEF doesn’t match OK.
      7. pf\_scan\_scaled vs pf\_stand: Chart, surface chart

         Description automatically generated  
         The standing book is not rotated against the original. We have a different perspective point, but brief handles it very well as we saw in the incline\_\* images.  
         When we compare an image to itself we found a perfect match between the features:  
         Graphical user interface

         Description automatically generated
      8. As we saw in the comparison of cheickenbroth images 01 and 04 the BRIEF performed well even under changes of illumination. This is so, since we calculate the R value in DoG filtering as a ratio between trace^2 and det of the gradient’s covariance matrix. Thus, as we saw in lecture, it will be invariant to scale and illumination (better in this sense from Harris Detector). Also, multiplying each pixel in the image (and also adding a bias to the pixel BTW) won’t change the relation between I(x) < I(y). it will preserve the same sign as we wish under illumination changes.
      9. We can relate to the scale in which the feature was detected and get a patchWidth that is scaled to that scale (not only 9x9 for all scales). This is what’s done in SIFT and other similar approaches.