GLOBAL TRANSFORMATION ESTIMATION VIA LOCAL REGION CONSENSUS

Erez Farhan

March 9, 2014

Ben-Gurion University

Abstract

This work presents a novel method for accurate estimation of perspective transformations in images. Recent publications have shown that considering a rich geometric transformation model of salient local regions, is extremely beneficial for the purpose of point matching.Yet, those methods are not used extensively for two reasons. First, because they are computationally more demanding, second reason is that the estimation of the geometric transformations are of limited accuracy and therefore the usage of them is limited. Moreover, in typical scenes, significant portions of the scene are low-textured and cannot be considered salient. Thus their transformation between images can be recovered only by related global transformations. It has been shown that considering a few local point matches in consensus can be very useful in estimating global transformation and rejecting outliers. Though these methods do not typically utilize information from local region transformations. In this work, we propose a mechanism that forges a consensus between local region correspondences and their underlying geometry , that allows the estimation of richer global transformations. We demonstrate how this idea enables the estimation of perspective transformations of planes, and locating many new point matches on practically texture-less areas of the scene.

1 Introduction

One of the key variabilities in the appearance of an image is its underlying geometry. Images of the same scene from different angles, may look very different. For the purpose of point matching, which is the process of finding corresponding scene points in two images, this variability has been intensively explored in the last decade [1, 2, 3, 4], specifically regarding local patches. For many applications, there are sufficiently good solutions in terms of the amount of matches, the portion of correct matches, their accuracy in locations and the computational demand. But, there are many other applications requiring improvement in on or more of these aspects.

The geometric variability of a local patch can be approximated as affine transformation. All the popular point matching methods [2, 4, 1] take into account some geometric variations, while accounting for the full affine model has proved to be beneficial [3, 1]. However, trying to estimate the geometric transformation from small patches is of limited accuracy, insufficient in many cases, and computationally demanding. The key difficulty in estimating affine transformation from small regions lie in the

Figure 1: The global deformation of the railroad is strongly projective (rectangle becomes trapezoid), while the local deformation of each tie is well approximated by affine transformation (rectangle becomes a parallelogram)

insufficiency and low quality of available information. The flip side of the local affine approximation, is that estimating richer transformations, such as projective transformations of planes, cannot be accurately estimated from small regions.
In this work, we show how forging a consensus between several local affine transformations of regions that share the same scene plane, can be utilized to improve the local transformation estimation and also allow accurate estimation of perspective transformation of the plane.

1.1 Affine And Perspective Transformations

Assuming that the patches around the corresponding points lie entirely on a planar surface of the scene, the geometric transformation between image patches can be described by a homography. This creates a perspective transformation of the image patch around each point match. In small enough region, the perspective transformation can effectively be approximated by an affine transformation. This fact was explored in many previous research works. Taking this fact into account proved to be extremely beneficial in finding point correspondences[3, 1]. In figure (1) we see the local tangency of perspective deformations to affine deformations. That is why the affine transformation estimation from local regions have focuses many research interest.
In [1], a method to adaptively normalize the affine shape of every point match is suggested. In [4] a method to normalize the affine transformation of detected region is suggested. In [3], a method to simulate the space of affine transformations is suggested. These methods show considerable improvement in terms of accuracy in position and performance when compared to methods that neglect these geometric variations [2, 5].
We assume that we are provided region matches given by MSER after affine normalization. The normalization is typically done using the measurements of the first and second order moments, to bring all detected regions to a canonical reference frame, up to an unknown rotation which is solved using standard methods [2]. An initial estimation of the affine transformation of a region enables predicting the corresponding locations of neighboring points that shares the same plane. The expected prediction

error grows lineally with the distance of the predicted point from the initial region, and is inversely proportional to the size of the initial region. We demonstrate how such a prediction can be done mutually between neighboring regions, and enables estimating the transformation of the union of the regions. The union dramatically increases the size of the estimated region, thus improving the estimation. It is also more likely to undergo significant perspective transformations, which can now be estimated to predict point matches in low textured areas of the plane, that are well beyond the initial salient regions. Another advantage of estimating from a union of regions, is that it is significantly less probable to encounter outlier matches between very large regions, and thus - this process can also be consider as an outlier rejection mechanism.

2 Estimation from Region Consensus

The process of estimating affine transformation between small image patches enables predicting the locations of other local regions in their surroundings. We use [4] as an initializer for the method, the output of this method is a list of region matches together with initial estimate of affine transformation, we assume this list includes large amount of false alarm and a rough estimate of the affine transformation. These matches will be called *phase zero* region matches. Given a *phase zero* match, the affine transformation $\hat{H}$ between the matching image patches can be estimated[6]. Using the affine transformation of each region, we can start looking for mutual relations between proximate regions. The transformation of every *phase zero* region is used to predict the transformation of neighboring regions, and then be validated using the estimated transformations of these neighboring regions. This is done as follows:

1. From each region in the source image [call it anchor region], we try to estimate the pixel locations of other regions in the target image, using the region's local transform estimation.
2. Since local transformation are not expected to be accurate on farther regions, even if lying on the same plane, we consider the distance between the current region to predicted regions in order to adjust the expected prediction errors.
3. The expected prediction error is also adjusted to the extent of the current region.
4. Using this knowledge and an assumption of the typical affine estimation errors of local regions, we can determine a maximum expected prediction error in estimating the locations of neighboring patches in the target image.
5. Once we have an expected error, we compare the predicted location with the location derived from the *phase zero* match of the neighboring region.
6. This comparison gives an estimate of the prediction error which is normalized by the expected prediction error to get the normalized prediction error.
7. The same process is done vise verse, with the neighboring region predicting the location of the anchor region in the target image. Yielding another normalized prediction error.
8. The maximum value of the two normalized prediction errors is then used as an agreement measure between the regions. If it is smaller than a fixed threshold, we consider both regions to have a co-planar consensus, and we unify them to one bigger *phase one* region.

(a) From anchor (green) to neighbor (red)

(b) From neighbor (green) to anchor (red)

Figure 2: Forging a consensus between a pair of regions. Blue axes represent the coordinate frame and extent of the estimating region. The black line represents the distance to the most difficult point to predict. The little red lines represent the predicted error factor in both axes

9. This process is repeated until we cannot find any new consensus pairs.
10. Finally, resulted *phase zero regions* that does not meet the required extent demands for estimating projective transformation are discarded.
11. We should note that outlier *phase zero* region matches are much less likely to forge a consensus with neighboring regions. This prevents these regions from being considered as a part of bigger union region, and are thus naturally discarded (phase 10).

The following sub-sections will elaborate on the more interesting phases mentioned above.

2.1 The effect of prediction distance and initial region size

Since affine transformations are mostly linear (neglecting the effect of translation), we can expect that the location prediction error using an estimated affine transformation will grow almost linearly with the distance of the neighboring point from the anchor region in the source image. Another theoretic notion is that the size of the anchor region also linearly affects the expected error of the estimated location of the neighboring region in the target image. in figure (2) we can see an example of the region consensus procedure. The important measures are the extent of the anchor region in it's coordinate frame (in dotted blue), and the distance from the anchor centroid to the most difficult neighbor point to predict (a black line). The most difficult point to predict is determined according to the formula:

$$\underset{p\in\Omega_{neighbor}}{\arg\max} \left\| (\|V_{anchor}\|^{-2} < p - \mu_{anchor}, V_{anchor} >, \|U_{anchor}\|^{-2} < p - \mu_{anchor}, U_{anchor} >) \right\|_2 \quad (1)$$

Where V, U are the un-normalized axes of the anchor region, and μ is the centroid point. Notice how this expression quantifies the projection of the translation between the centroid and the point p, on the different axes. Each projection is multiplied with the inverse of the norm for axis normalization. finally, each projection is multiplied once again in the inverse of axis norm to account for the importance of region extent in that axis. i.e. the wider the region is in a certain axis, the easier it would be to predict the transformation of point along that axis.

(a) Predicted neighbor (green), neighbor given by *phase zero match* (red)

(b) Predicted anchor (green), anchor given by *phase zero match* (red)

Figure 3: Evaluating a consensus between a pair of regions in the target image.

equation (1) also introduces the notion of *normalized expected prediction error*. When we find the point p that achieves the maximum, we can now know what error factor to expect from the location prediction. Using this factor, we can decide on the feasibility of any candidate consensus.

2.2 Forging a consensus and perspective estimation

After introducing the notion of *normalized expected prediction error,* all we have to do decide if a pair of anchor and neighbor region can be forged in a consensus, is to measure the mutual prediction errors in the target image, normalize it, and threshold on some determined fixed threshold. In figure(3), we can see the mutual prediction results of the example given in (2). The prediction error is very visual here. In a perfect consensus case, the red and the green regions should sit perfectly on top of each other. Although the results in this example are far from perfect, this consensus is actually approved, since after error normalization, this error goes below the fixed threshold. In the next phase, we see that this decision was very helpful.

Like mentioned, if a pair of regions forge a consensus with normalized prediction error above a certain threshold, they are unified to create a bigger region. in figure (4), we can see how the exact same process is extended for the unified region, only now the expected prediction error for prediction the neighbor using the anchor is significantly smaller - since the anchor region is bigger. This results in a much more accurate target prediction of the new neighbor, as we can see in figure (5). We should note that any region can be added only to one union, this saves computational time and ambiguities, in the expense of possible in-optimality. Finally, after applying this process on the group of all regions, we have unions of regions that appear to be co-planar and to agree on local transformation. We can use these unions to estimate the perspective transformation of the union from the source frame to the target frame. This can be done very accurately, since the extent of the unions is very large, and thus the expected error very small.

(a) From anchor (green) to neighbor (red)

(b) From neighbor (green) to anchor (red)

Figure 4: Forging a consensus between a pair of regions, when the anchor region is already a unified region of two *phase zero* regions. Prediction of neighbor target location is made much easier

(a) Predicted neighbor (green), neighbor given by *phase zero match* (red)

(b) Predicted anchor (green), anchor given by *phase zero match* (red)

Figure 5: Evaluating a consensus between a pair of regions in the target image.

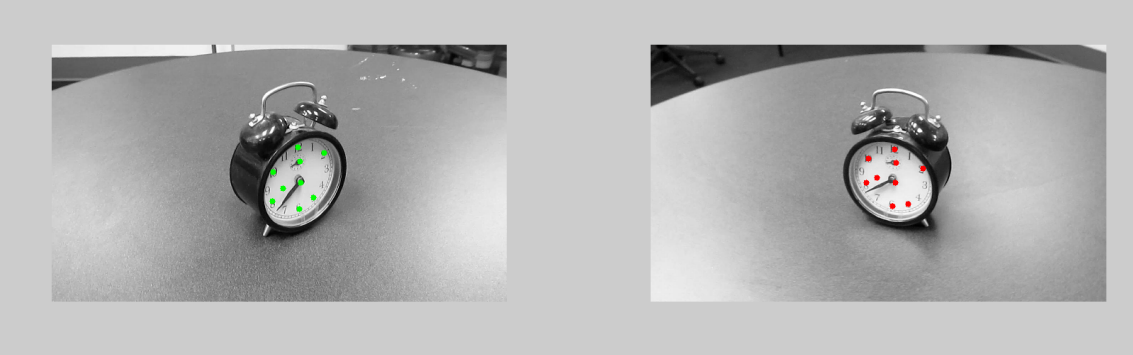

Figure 6: Typical result of the application

2.3 Post process: rejecting "lonely" regions

The process described in sub-section(2.2), may leave some of the regions outside any union. These "lonely" regions are more likely to be coming from false region matches, since they couldn't found any neighboring regions assessing their underlying transformation estimation. Of course, some "true" matches may fail to find consensus, and thus rejected by this mechanism. In this manner, the proposed algorithm acts as an outlier filter.

3 Application: Matches of Arbitrary Points on the Plane

Since we have collected many co-planar local transformations to a more global transformation of the plane, we can now predict the target image locations of arbitrary points in the source image that lie on plane with now attached perspective transform estimation. The problem now is to determine which estimated perspective transform is appropriate for applying on the desired point. Different unified regions may correspond to different scene planes, so the choice of the appropriate region and thus perspective transform may be crucial. In this work, we make a reduction, that the scene has only one highly-textured plane, and thus all unified groups, as well as the desired point, lie on the same plane. This reduces the decision to choosing the most appropriate *phase one region* with its underlying transform estimation, to predict the desired point location in the target image. For this task, again the notion of *normalized expected prediction error comes to aid.* We choose the region that has the smallest *normalized expected prediction error* in predicting the target location of desired point. Naturally, this results in regions that situate relatively close to the desired point in the source image, and have a long extent in the direction of the translation vector between the region and the desired point.
To illustrate this ability, a MATLAB application has been created, in which 2 images of a mostly planar scene can be loaded, and correspondences of arbitrary points on that plane can be found.

3.1 How to use the Matlab application

The use of the dedicated GUI is pretty straight forward, here are the standard steps:

1. Extract the code package, enter its root directory in matlab, and run the script 'setup.m'.

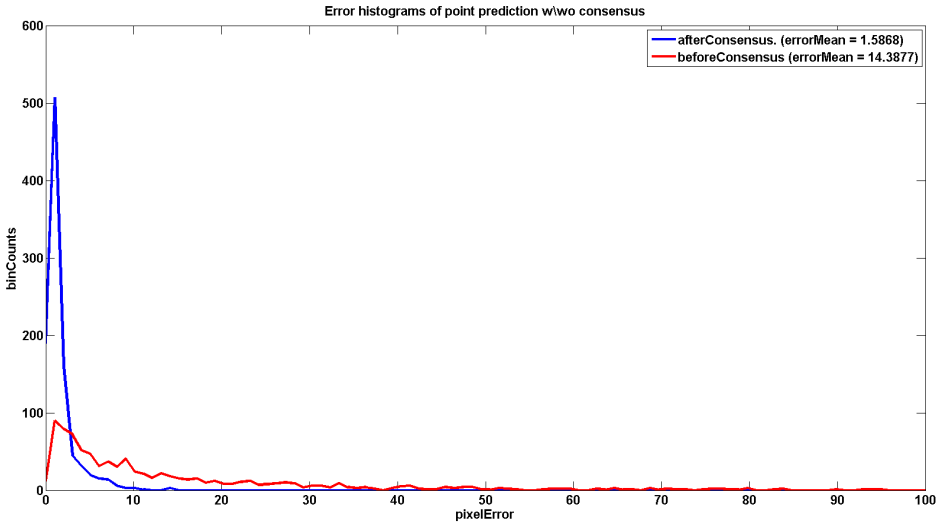

Figure 7: The contribution of the consensus mechanism to arbitrary Point Matches Performance

2. Run 'ConsensusGui.m'. A simple GUI should be opened. Notice that 'Try it!' button is initially deactivated.
3. Now hit 'Load Source Image' and choose a source image to work with. This will do some process (finding *phase zero regions*), so be patient.
4. Same goes for hitting 'Load Target Image'.
5. The two images should now appear in one figure along side each other.
6. Now press process - this runs the actual algorithm proposed here.
7. Now 'Try it!' button is enabled, press it. Now you need to pinch a desired point in the left image. The corresponding point immediately appears in the right image.
8. Hit 'Try it!' again to pinch another point.
9. If you want do load different images, make sure you hit process after loading the images.

In figure (6), we can see a typical result of the application after pinching a few points.

4 Test and Results

We have checked the performance of the point matching application on several standard feature matching benchmark images given in[7]. To verify the contribution of the proposed consensus algorithm, for each source image of the benchmark, we have sampled 100 random points and tried to predict it's location in the target image. Since we have the ground truth homography from [7], we can measure the

pixel error in the target image. We compare this error to the error produced by predicting the target point using the collection of *phase zero region matches* in the same manner described in section (3). We collect all the results from all the 10 benchmark image challenges that introduce viewpoint changes ('graffity' and 'wall') and present the pixel error histogram of both methods in figure (7). Notice that error above 100 pixels were discarded, for a more convenient presentation .Errors above 100 pixels were seen only in the method without consensus, if we were to regard them, the mean error of the methods without consensus would have been above 66 pixels. It is clearly evident how the consensus mechanism prevents cases of very big error [all errors below 15 pixels], and significantly improves the overall accuracy of arbitrary point predictions.

5 Summary and future ideas

The ability to estimate the image transformations accurately, using a simple existing technique as initializer and then considering region consensus, enabled us to accurately predict the location of arbitrary points in the target image. We saw how the consensus mechanism increased accuracy and enabled the consideration of more rich transformations like the perspective transform. The mechanism also contributed for rejecting large prediction errors. The work here was under a reduction to mostly planar scenes, but the idea can be certainly extended to more complicated situations, and richer transformations. An immediate future development would be to detect situations of different multiple planes, and find an accurate consensus to every plane. This idea could also be accustomed to elastic transformations, tough possibly with some algorithmic variations regarding the estimation model.

References

[1] Krystian Mikolajczyk and Cordelia Schmid, "Scale & affine invariant interest point detectors," *International journal of computer vision*, vol. 60, no. 1, pp. 63–86, 2004.

[2] David G Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] Jean-Michel Morel and Guoshen Yu, "Asift: A new framework for fully affine invariant image comparison," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 438–469, 2009.

[4] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*, pp. 404–417. Springer, 2006.

[6] Jacob Bentolila and Joseph M Francos, "Affine consistency graphs for image representation and elastic matching," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 2365–2368.

[7] Krystian Mikolajczyk, "Affine covariant features," `http://www.robots.ox.ac.uk/~vgg/research/affine/index.html`, Accessed: 2014-01-29.