**Better Title: Finding Matches Between Two Images**

**Yoni Nachmany, United States of America; Daniil Vodopian, Russia**

**Mentor: Inbar Aharon**

**Supervisor: Dr. Yaron Lipman**

**Dept. of Computer Science and Applied Mathematics**

**Abstract:**

Up to 150 words

single spaced

abbreviations:

Scale Invariant Feature Transform (SIFT)  
Random Sample Consensus (RANSAC) algorithm

In computer vision, identifying corresponding pairs of points in two images is crucial for feature matching tasks such as facial recognition and satellite tracking. We did a, b, c, d for each algorithm/program...

**Introduction:**

Finding the most accurate transformation between two subsequent images is a fundamental challenge in computer vision. Scale Invariant Feature Transformation (SIFT) is an algorithm that returns pairs of matching points in three steps. First, SIFT finds points in the original image that arestable for geometric transformations, using the Difference of Gaussian (DOG) method. For each stable point, SIFT calculates a “descriptor,” a vector of 128 attributes derived from the Histogram of Gradient (HOG) that describes the point. Lastly, SIFT identifies points in the second image that match points in the first image, by finding the point with the closest descriptor.

With the pairs of matching points that SIFT returns, the Random Sample Consensus (RANSAC) algorithm finds the transformation between the pairs of points, simultaneously dropping outliers from SIFT’s results. Over many iterations, RANSAC randomly samples three points at a time and calculates the affine transformation for these points, keeping track of the best affine transformation.

The other option for finding the best transformation is the Iteratively Reweighted Least Squares (IRLS) algorithm. Whereas RANSAC takes only a subset of points, IRLS looks for all the affine transformations. Compared to RANSAC, IRLS is more accurate but more computationally intensive. **In our experience, the tradeoff for greater accuracy was (not) worth the increased time of calculation.**

**Methods/Experimental:**

The programming language used throughout this project was Matlab Version 7.11. In the code we used the VLFeat library for the SIFT computer vision algorithm.

SIFT returns pairs of points (from the source image) and (from the target image). Our goal was to identify the transformation **T** that maximizes the match between **T(P)** (the result of transformation **T** on matrix **P**) and , i.e finding transformation **T** that minimizes The power of 0 means that if the value is 0, then the function is also 0, otherwise the function is 1.In our project, we want to find an affine transformation such that , where is the transformation matrix combining scaling and rotation and is the translation vector. We can insert both the transformation A and the translation t into a single matrix, of the form

**RANSAC**

Our RANSAC function takes as arguments set of points P and Q, as well as the number of iterations and the threshold. The iterations refer to how many set of triples are tested to find the most accurate transformation T. The threshold is the ratio of the distance between the best matching and the second best one, with a default value of 1.5 in SIFT. (YOU EXPLAIN HERE THE THRESHOLD OF SIFT AND NOT FOR RANSAC!)

Based on the threshold, RANSAC identifies inliers for each transformation and keeps track of which transformation returns the most inliers. RANSAC provides the best transformation matrix T and the set of inliers.

**IRLS**

Our IRLS function…FOR LATER!!!

**Results:**

Using MATLAB’s graphing functions, we plotted the source and destination images, with two copies of each image side-by-side. On the left side, the points are the inliers within the set of P, and on the right side are the corresponding matches in Q on the right image. The black lines on the right side represent the distance from Q to TxP.

On the two images we can see the result of changing the threshold for the RANSAC method. The threshold of the first figure is 3, while on the second one it’s 10. So the bigger the threshold is, the more points match the transformation, and the more inliers we have on the images. For instance, threshold of 3 is not big enough to detect the head of Indiana Jones (on the right), while threshold of 10 lets us see the matches on his face. However, even the threshold of 10 isn’t enough to detect the head of Marcus Brody (on the left). (YOU DIDN’T TALK ABOUT THE OTHER SIDE-THE BIGGER THE THRESHOLD IS, THE LESS ACCURATE THE TRANSFORMATION)





bestT1 =

1.0061 0.0044 14.3802

0.0088 0.9806 -1.2897

0 0 1.0000

-(before) SIFT has many inliers

nearly the identity matix [1 0; 0 1] plus small translation (IN CUP, larger transformation and matrix)

bestT2 =

0.9637 0.1175 9.1993

-0.0054 0.9935 0.6109

0 0 1.0000



**Discussion:**

1.5

spaced

CONCLUSION, LOOKING FORWARD, WHAT CAN BE DONE NEXT? DIFFERENCE BETWEEN DIFFERENT THRESHOLD (some change) AND ITERATIONS (no real change)

**Acknowledgements:**

1.5

spaced

link to Github

**References**

1. Distinctive image features from scale-invariant key points...

2. Random Sample Consensus...

3. A Tutorial on MM Algorithms...

4. iteratively reweighted least squares...