Abstract

Image registration is one of the most important steps in most Image Processing tasks for which the final result is achieved from a combination of various resources. Automatic registration of remote-sensing images is a difficult task as it must deal with the intensity changes and variation of scale, rotation and illumination of the images. Our project proposes image registration technique of multi-view, multi- temporal and multispectral remote sensing images. Firstly, a pre-processing step is performed by applying Median Filtering and Histogram Equalization to enhance the images. Secondly, the Steerable Pyramid Transform is adopted to produce multi-resolution levels of reference and sensed images; then, the AKAZE (Accelerated KAZE) algorithm is utilized for extracting feature points that can deal with the large variations of scale, rotation and illumination between images. Thirdly, matching the features points by using the Euclidian distance ratio; then removing the false matching pairs, also known as outliers, using our own version of improved RANdom SAmple Consensus (RANSAC) algorithm. Finally, the mapping function is obtained by the affine transformation using BF (Brute Force) algorithm. The complete application is based on Python language and is based on OpenCV libraries and GDAL libraries as well. For the Graphical User Interface, PyQT4 has been used, since it is an open source software.

Quantitative comparisons of our technique with the related techniques show a significant improvement in the presence of large scale, rotation changes, and the intensity changes. The effectiveness of the proposed technique is demonstrated by the experimental results.

2. Scope and Limitations

Automatic image registration has been widely studied in the fields of computer vision, remote sensing, and medical imaging. In various cases, such as image fusion, high registration accuracy should be achieved to meet application requirements.

Using manual geo-registration, each set of images (input and reference image) are processed separately under human supervision. This takes a lot of time and efforts as well. Our project addresses this and we aim at creating an auto registration tool which processes an array of set of images, and gives us a set of geo-referenced images. The project can be further applied to geo-reference drone imagery.

However, since the algorithms used are very resource intensive, the program needs at least 8GB of RAM to run. Due to time limitations we were not able to achieve many to one geo-referencing of images, also because of paucity of time we were not able to expand our projection system to support WGS (World Geodetic System) and UTM (Universal Transverse Mercator).

e Project

Image registration is the process of overlaying images of the same scene taken at different times, from different viewpoints, and/or by different sensors. In other words it is a process of finding a transformation that aligns one image to another. The image which is registered is called the reference image and the one which is to be matched to the reference image is called the sensed image.

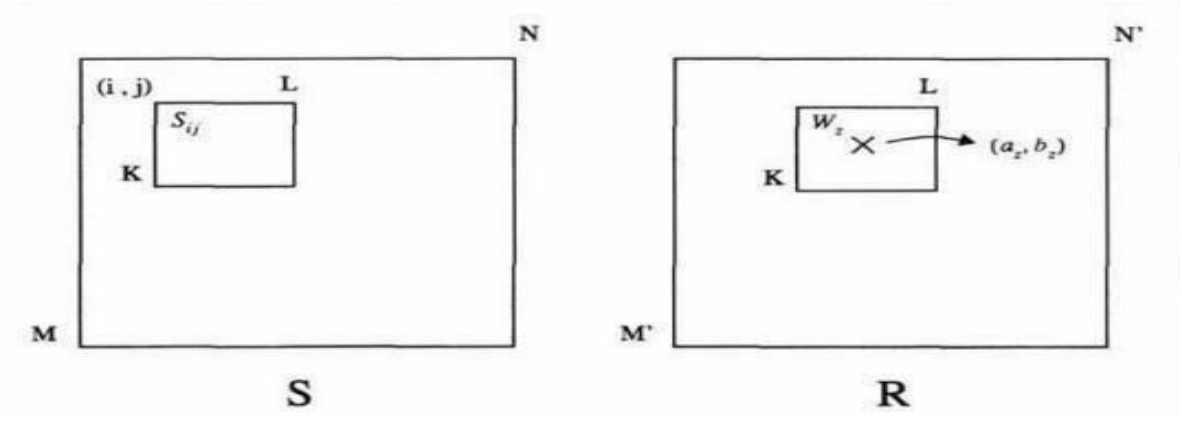
Image registration is the most critical operation in remote sensing applications to enable location based referencing and analysis of earth features. Automatic registration of remote-sensing images is a difficult task as it must deal with the intensity changes and variation of scale, rotation and illumination of the images.

This project aims to perform batch-wise image registration using GDAL. At the same time we use open source software to achieve this[i]. Different algorithms are employed to achieve this, based on pattern recognition, feature based detection, similarity techniques etc. In general, the registration methods are different from each other in the sense that they can combine different techniques for feature identification, feature matching, and mapping functions. The most difficult step in image registration is obtaining the correspondence between the two sets of features. This task is crucial for the accuracy of image registration, and much effort has been spent in the development of efficient feature matching techniques. The traditional manual approach used human assistance to identifying the control points in the images. In this approach, the steps of feature identification and matching are done simultaneously. The images are displayed on the screen and the user chooses corresponding features in the images which clearly appear in both the images. Candidate features include lakes, rivers, coastlines, roads, or other such scene-dominant man-made or natural structures. Each of these features will be assigned one or more point locations (e.g., the centroid of the areas, or line endings, etc.), and these points are referred to as control points. These control points are then used in the determination of the mapping function. In order to get precise registration, a large number of control points must be selected across the whole image. This is a very tedious and repetitive task. Furthermore, this approach requires someone who possesses knowledge of the application domain and is not feasible in cases where there is a large amount of data. Thus, there is a need for automated techniques that require little or no operator supervision. Based on the nature of features used,

automated registration methods can be broadly classified into area-based and feature-based techniques:-

4.1. Area Based Registration

In the area-based methods, a small window of points in the reference image is statistically compared with windows of the same size in the sensed image. The process of Area Based Registration is shown in Figure 1 below.



13.1. Fig. 1 Area based Registration

Area Based Method calculates a certain measurement using gray-data in the fixed-size window from two images and treats the center points of windows as corresponding points when the measured value exceeds the threshold.

The main disadvantage of Area Based Method lies in the excessive computing time spent on searching for the corresponding point.

eature Based Registration

In feature based method algorithms, salient features are extracted from different images, and then corresponding features are determined by comparing the similarities of their descriptions. Matching methods based on point feature, such as SIFT (Scale Invariant Feature Transform) and SURF (Speeded Up Robust Features), are widely used. Other methods based on features, including lines, edges, contours, and shapes, are also used in numerous applications. Figure 2 shown below demonstrates the feature based method of registration.

13.2. Fig. 2 Feature based Registration

The shortcoming of SIFT is excessive memory consumption. Another problem is that for extremely large remotely sensed images, the direct use of SIFT-based matching faces difficulties in obtaining evenly distributed corresponding points.

A basic procedure that we followed is as follows:

Firstly, the Scale Invariant Feature Transform (SIFT) is utilized for extracting feature points that can deal with the large variations of scale, rotation and illumination between images. Secondly, the feature points are matched by using the Euclidean

distance ratio, then removing the false matching pairs using the RANSAC (RANdom SAmple Consensus) algorithm.

13.3. Fig. 3 Basic Procedure

6. Methodology

Python was chosen as the preferred interpreter, given the availability of scientific and machine learning tools for it. To ensure cross platform GUI, PyQT4 was to be used, which is natively supported by Python. OpenCV will be used for Image Processing Tasks and GDAL (Geospatial Data Abstraction Library) will be used for Registration.

The following Algorithms were tested on actual satellite imagery data (15653x14506x3 and 9473x7983x4), and the results obtained on a system (Intel i7 4710HQ, 8GB RAM, NVIDIA GeForce GTX 960, Windows 10/Ubuntu 16.04) are described:

The project can be divided into the following parts: -

Image pre-processing

Find the GCPs by using suitable algorithms using feature detection Refine those tie points by removing outliers

Re-sampling

The Working of the program is as follows: -

6.1. Data Importing

First step is extracting information from the given images. Geospatial raster data is a heavily used product in Geographic Information Systems and Photogrammetry. Input and output images that we are using are represented using raster data. The standard library for loading GIS imagery is the GDAL. We import the GDAL Libraries using the command "**from osgeo import**

gdal".

First we open the raster so we can explore its attributes.. GDAL will detect the format if it can, and return a *gdal.Dataset* object.

ds = gdal.Open('your_Image.tif')

"array = ds.ReadAsArray()" can be used to read raster data.

6.2. Pre-Processing

After reading the raster data the data is stored as Numpy array as GDAL supports Numpy array. This is followed by normalization of pixel values between 0-255 or an 8 bit size.

The extraction of image data is followed by pre-processing of the image. Image pre-processing is performed using histogram equalization.

Original Image

13.5. Fig. 5 Histogram Equalization

Equalized Image

By using histogram equalization you get a much flatter histogram response. This actually increases the overall contrast of the image. Increasing the contrast makes the processing of the images fast and efficient as pixels can be easily differentiated after histogram equalization.

After this image pre-processing step the next is **feature detection.** Feature extraction is done to obtain tie points. A tie point is a point within the image that has the following characteristics:-

It has a clear, preferably mathematically well-founded,
definition, It has a well-defined position in image space,

The local image structure around the interest point is rich in terms of local information contents.

There are basically three aspects of our Methodology, viz., **Feature Detection**, **Descriptor Extraction, Descriptor matching**, **Outlier Rejection** and finally **Image Warping.**

6.3. Feature Detection: The following algorithms have been used in our program:

6.3.1. Harris Corner Detection for feature extraction and Cross-Correlation for Point

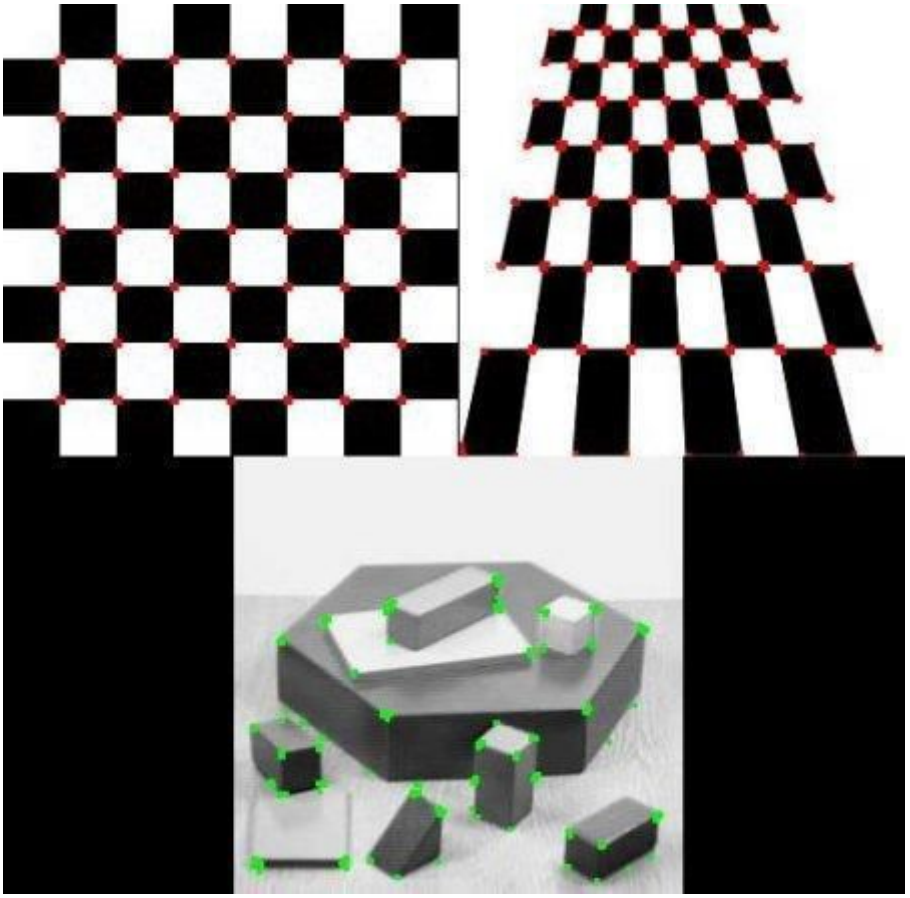
Detection: In this method, the image was split into 5x5 equal sized tiles and the most

prominent corner in each tile was classified as a feature. The corresponding point for each feature on the Reference Image was detected using Cross Correlation with a template image of 50 pixel padding around the feature detected.

6.3.1.1. Outcome: The algorithm is translation and rotation invariant, but is not scale invariant, and is highly susceptible to illumination differences. Also, it requires high processing powers, which are beyond the capacities of normal PC computers as of today.

Time taken to process entire image: 13 minutes

Time taken to process image resized by half: 6 minutes



13.6. Fig. 6 Harris Corner Detection Results

6.3.2. FAST (Features from Accelerated Segment Test) for Feature extraction

Correlation with Normed distance: FAST algorithm is based on a paper by Edward Rosten and Tom Drummond - "Machine learning for high-speed corner detection[ii]. For detection, Normed distance was used instead of Euclidean distance and Image was

preprocessed by histogram equalization before applying FAST to minimize differences due to illumination variation.

6.3.2.1. Outcome: The algorithm is translation and rotation invariant, and works for images across different illuminations, but is **NOT scale invariant**. It is reasonably fast, and does not require very high processing powers to compute. Time taken to process entire image: 100-110 sec

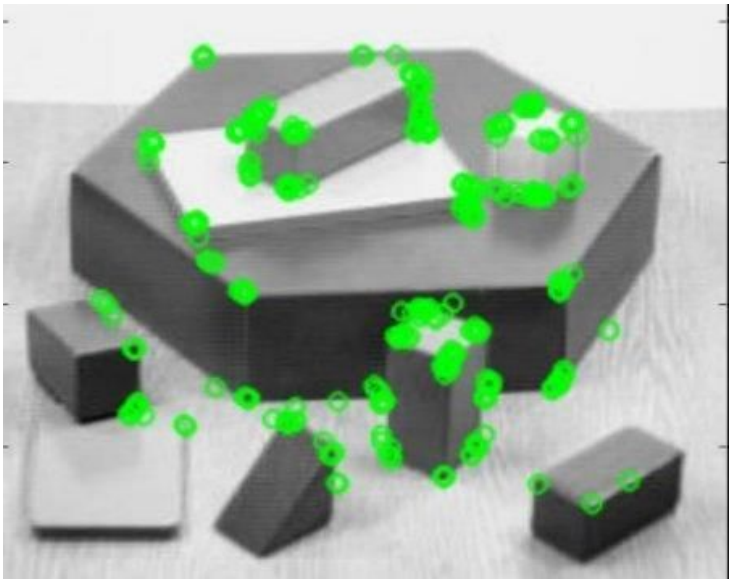
Time taken to process image resized by half: 50 sec

6.3.3. ORB (Oriented-fast and Brisk Rotation): This algorithm uses FAST algorithm for Feature Extraction on both images, and uses BRISK (Binary Robust Invariant Scalable Key points) to find relation between the found key points[iii]. BRISK is a much optimized algorithm. Image was preprocessed by histogram equalization before applying ORB to minimize differences due to Illumination variation.

6.3.3.1. Outcome: The algorithm is translation and rotation invariant, and works for images across different illuminations. The algorithm is **partially scale invariant,** and could handle variations in scale up to a factor of 2. It is very fast, and does not require very high processing powers to compute. This algorithm was found to be the best algorithm to calculate transformation if scale variance is within sufficient bounds.

Time taken to process entire image: 51 sec

Time taken to process image resized by half: 17 sec



13.7. Fig. 7 ORB Feature Detection Results

6.3.4. SIFT (Scale Invariant Feature Transform) for Feature Extraction and RANSAC (RANdom SAmpling Consensus): This algorithm uses SIFT algorithm for Feature Extraction on BOTH IMAGES, and uses RANSAC to find relation between the found key points[iv]. RANSAC is a training-based machine learning model. The image was pre-processed by histogram equalization before applying ORB to minimize the differences due to illumination variation.

6.3.4.1. Outcome: The algorithm is translation and rotation invariant, and works for images across different illuminations. The algorithm is fully scale invariant, which makes it different from all other algorithms. Calculating SIFT features directly is a very computationally expensive task, but it can be fastened by using SURF (Speeded-Up Robust Features) that reduces the computations to a good extent. This algorithm was found to work robustly even for satellite imagery. Hence, this

algorithm was chosen for further work.	5 minute
Time taken to process entire image (using SURF):	50 sec
Time taken to process image resized by half:	

13.8. Fig. 8-(i) SIFT Algorithm Results with no clouds present

13.8. Fig. 8-(ii) SIFT Algorithm Results with clouds present

13.8. Fig. 8-(ii) SIFT Algorithm Results without RANSAC

6.3.5. BRIEF (Binary Robust Independent Elementary Features): BRIEF is a faster method feature descriptor calculation and matching. It also provides high recognition rate unless there is large in-plane rotation

6.3.6. BRISK (Binary Robust Invariant Scalable Keypoints): BRISK relies on an easily configurable circular sampling pattern from which it computes brightness comparisons to form a binary descriptor string. The unique properties of BRISK can be useful for a wide spectrum of applications, in particular for tasks with hard real-time constraints or limited computation power: BRISK finally offers the quality

of high-end features in such time-demanding applications. BRISK achieves comparable quality of matching at much less computation time.

13.9 Fig. 9 Feature Matching Using BRISK.

6.3.7. AGAST (Adaptive and Generic Corner Detection Based on the Accelerated Segment Test): This algorithm finds the optimal decision tree in an extended configuration space which can be combined to yield an adaptive and generic accelerated segment test. The resulting method provides high performance for arbitrary environments and so unlike FAST, the corner detector does not have to be trained for a specific scene, but it dynamically adapts to the environment while processing an image.

6.3.8. A-KAZE (Accelerated KAZE): KAZE is an edge-preserving non-linear filtering strategy to locate image features. The filtering is based on the image diffusion equation. A fast explicit diffusion (FED) technique is used to solve the diffusion equation efficiently, contributing to an accelerated variation of the KAZE algorithm, called AKAZE. Keypoints are located by finding the extrema of the second-order derivatives of the image over the non-linear multi-scale pyramid built from the principle of image diffusion.

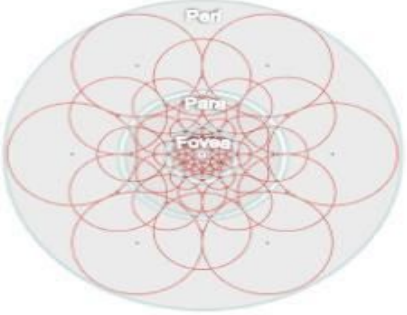
A-KAZE deploys a technique similar to SURF to estimate the direction of a patch. A modified local difference binary (LDB) representation of the rotation-compensated patch is then extracted as its binary descriptor.

13.10. Fig. 10 AKAZE Feature Detection Results

6.4. Descriptor Extraction

After detecting key points we go on to compute a descriptor for every one of them. A local descriptor is a compact representation of a point's local neighborhood. In contrast to global descriptors, describing a complete object or point cloud, local descriptors try to resemble shape and appearance only in a local neighborhood around a point and thus are very suitable for representing it in terms of matching. **OpenCV** comes with several implementations for feature detection. The algorithms used by us include **SIFT, BRISK, FREAK, AKAZE.**

6.4.1. FREAK (Fast REtinA Keypoint): FREAK suggests using the retinal sampling grid which is also circular with the difference of having higher density of points near the center. The density of points drops exponentially. Each sampling point is smoothed with a gaussian kernel where the radius of the circle illustrates the size of the standard deviation of the kernel.



13.11. Fig. 11 Working of FREAK Algorithm

6.5. Correspondence Estimation (Descriptor Matcher)

The next task is to find correspondences between the key points found in both images. Therefore the extracted features are placed in a structure that can be searched efficiently. Usually it is sufficient to look up all local feature-descriptors and match each one of them to its corresponding counterpart from the other image. However due to the fact that two images from a similar scene don't necessarily have the same number of feature-descriptors as one cloud can have more data than the other, we need to run a separate correspondence rejection process. The following algorithms have been taken into use for our program: -

6.5.1. BF (Brute Force): It takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation. And the closest one is returned.

13.12. Fig. 12 Brute Force Algorithm Results

6.5.2. FLANN (Fast Library for Approximate Nearest Neighbors): FLANN is a library for performing fast approximate nearest neighbor searches in high dimensional spaces. It contains a collection of algorithms we found to work best for nearest neighbor search and a system for automatically choosing the best algorithm and optimum parameters depending on the dataset.

13.13. Fig. 13 FLANN Algorithm Results

6.6. Outliers Rejection

The next step is **outliers rejection** or rejecting those points which exceed the threshold of the RMSE (Root Mean Square Error) value. One of the most common approaches to perform correspondence rejection is to use RANSAC (Random Sample Consensus).

6.6.1. RAndom SAmple Consensus (RANSAC): The RANSAC algorithm is a learning technique to estimate parameters of a model by random sampling of observed data. Given a dataset whose data elements contain both inliers and outliers, RANSAC uses the voting scheme to find the optimal fitting result. Data elements in the dataset are used to vote for one or multiple models. The implementation of this voting scheme is based on two assumptions: that the noisy features will not vote consistently for any single model (few outliers) and there are enough features to agree on a good model.

6.7. Image Warping

Changing raster format is done using **gdal_translate.** Suppose you have a raster in UTM coordinates but it is not in .flt format. One can change the format using gdal_translate.

Using **gdalwarp** we can specify output format for the transformation polynomial order and the interpolation.

7. Comparison of Applied Algorithms

	Translation Invariant	Rotation Invariant	Scale Invariant	Illumination Invariant	Process Time* Full Image	Process Time* Half Image
Harris Corner +Correlation	Yes	Partially	No	No	13min	6min
FAST +Correlation*	Yes	Yes	No	Yes	110sec	50sec
ORB + FLANN	Yes	Yes	Partially	Yes	51sec	17sec
SURF + RANSAC	Yes	Yes	Yes	Yes	5min	50sec
AGAST + FREAK	Yes	Yes	Yes	Yes	7min	2min
BRISK	Yes	Yes	Yes	Yes	6.5min	2min
AKAZE	Yes	Yes	Yes	Yes	3min	40sec

13.14 Fig. 14 Comparative Study of the Algorithms used

9. FLOWCHART

Figure shows the Image processing procedure which is followed in the background when the program is run.