Improving Data Driven Image Geolocation

Paul Sastrasinh (psastras@cs.brown.edu)

1 Introduction

Given a photograph, can you determine where in the world it was taken? Depending on the photograph, this may be a very easy or difficult question. Given a picture of the Eiffel tower, for example, most people would be able to correctly determine that the photograph was taken in Paris, France.

Hays and Efros [3] proposed an algorithm for automatically estimating where on the earth an image was taken by using a mostly data driven, scene matching approach. Given an input photograph, the system matches that photo against all the other scenes in the database. Their database consists of about 6.5 million geotagged images collected from Flickr. The system relies very little on complex scene understanding and instead uses large amounts of data in the hope that given an image, the system can find a close match that image to some photos in the database. Once the best image matches are found, the matches are clustered into geographically and the best cluster is chosen as the estimated location.

Even utilizing large amounts of data, im2gps is only able to correctly identify the location of a photo to within 200km of its actual location with a success rate of only about twenty percent. Arguably, there may be some images which are impossible to accurately geolocate[1], so it is difficult to determine an upper bound in accuracy, but there is certainly much room for improvement. It is our belief that computers should be able to outperform humans at this task. In particular, we seek to improve the performance of im2gps through better data and scene understanding through stronger features.

2 Building a Larger Dataset

In the original im2gps, the system uses a database of approximately six million geotagged images. It is unsurprising that im2gps performance depends heavily on the database of images that it can

[1] For example very similar pictures of tropical beaches could occur in Hawaii, Thailand, the Caribbean, etc.

match against. Given a novel photograph with no similar images in the database, the system has no hope of trying to correctly identify where that image, since it makes no attempts to parse or understand the image. As more images are added to the database, we should expect an increase in performance up to some threshold where we have sampled the space of images completely or our features are insufficient to distinguish between images. Hays and Efros [3] show that im2gps performance strictly increases given a larger database up until 6.35 million images (the full size of their database).

2.1 Database Collection

To take advantage the predicted increase in performance with database size, we collect a new set of images from Flickr, using publicly available geotagged images uploaded from 2001 through 2011. Like im2gps, these images are then filtered by location keywords (ie. America, Thailand), negative keywords (ie. wedding, baby), and size (at least 1024 pixels in resolution in either dimension). Images with a geotag accuracy of less than six are also filtered out[2].

The original im2gps collected in 2007 is made up of 6.35 million images after filtering from a set of about 20 million geotagged images. At the time of data collection (2011), the number of publicly available geo-tagged images on Flickr was about 160 million. This increase in number of images shows an ever increasing number of images being made available on Flickr (and we expect similar trends for other photo sharing sites - see Figure 2.1). While data collection is still ongoing with about 8TB of data collected so far, we expect the final database size after filtering to constitute about 60 million[3] images and 10TB of data, which is a ten-fold increase over the previous data set.

The positive keywords account for a significant portion of the geo tagged data on Flickr, but there are an increasing number of photos not being captured by the selected positive location keywords. Nevertheless, using these positive keywords makes it more likely to collect photos which are useful for geolocation, since many users will upload geotagged photos which are not relevant for geolocation (ie. pets or random objects). The positive keyword list used here is the same as in the original im2gps[4]. Even using both positive keywords and negative keywords, the database still contains many images which are not visually relevant (see Figure 2.1). Negative keywords were the same as the ones used for im2gps, with more added on. See the Appendix for more information.

Even with this large increase in data, this averages out to only about 0.4 images per square kilometer of the earth's landmass (compared with 0.0435 for the original im2gps database), which is still far from an exhaustive sampling of the earth. As shown in Figures 2.12.1, the data is in fact far from a uniform sampling of the earth and instead tends to be clustered around popular destinations and cities.

[2] A geotag accuracy of less than six indicates GPS accuracy of less than regional level (ie. world or country) according to Flickr.

[3] Note that the chart in Figure 2.1 shows the total number queried images after filtering by positive and negative queries, but not by resolution. We do not know the image resolution until it has been downloaded, making it difficult to exactly determine how many images meet our resolution criteria before downloading. Even though there are ~70 million queried images after keyword filtering, the final number of downloaded images is expected to be somewhat fewer.

[4] This includes every continent, country and territory, U.S states, the top 200 cities, and popular tourist sites.

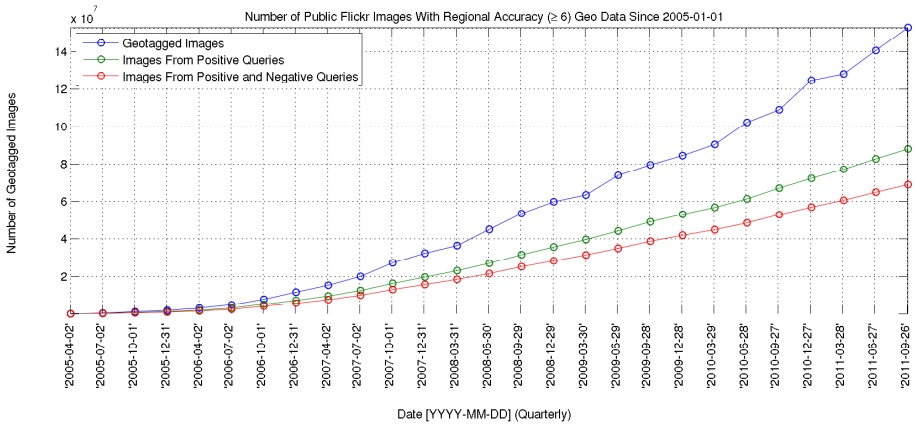

Figure 1: Plot of total number of geotagged Flickr images with regional accuracy or higher. Geotagged images counts the total number of publicly available geotagged images on Flickr. Images from positive queries counts those images which are captured using our input location queries (ie. America, Thailand ...). Images from positive queries counts the number of images after filtering with both positive and negative queries.

Figure 2: Sample images from the new database. All these images were found to have the keyword "Taipei," yet some of them are not useful for geolocation.

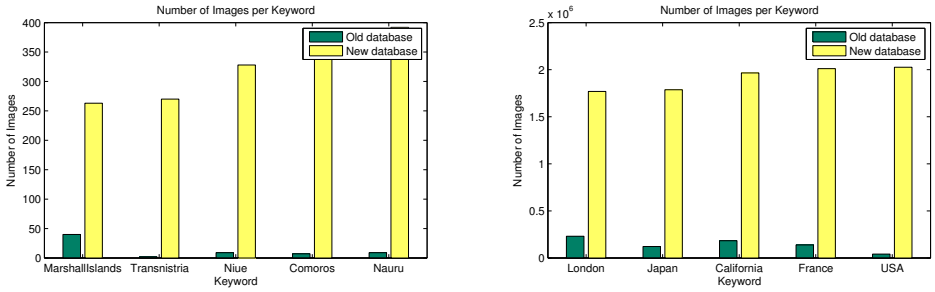

Figure 3: Graphs showing the number of images in the old versus the new database for the bottom and top five keywords sorted by the total difference in images (#new images - #old images). Note that keywords which already had a relatively large number of images (ie. USA, France, California) show the largest increase in new images.

Examining the new database compared to the old database shows that some of the largest increase in number of photographs occurs in areas which were already densely sampled in the previous dataset. This suggests that as we collect more data in the future, the distribution of photos may become even more biased towards certain areas. Even so, we expect an increase in geolocation performance with this increase in data.

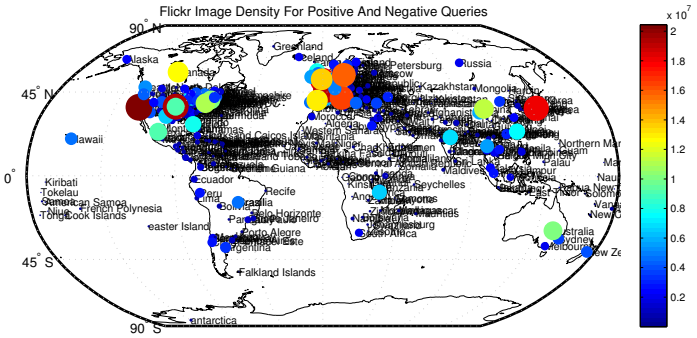

Figure 4: Plot showing the total number of images for each positive keyword (after being filtered by negative keywords). Unsurprisingly, parts of North America, Europe, and Japan tend to have the highest number of images available. Note that the scale and color of the circles represents the relative number of images.

Once the database collection is complete, im2gps features will be built for each image and added to the database.

3 What Features Work Well?

It has been shown [3] that im2gps performance varies among scene type. Landmarks, for instance, are most visually distinctive and can often be unique recognized. They are also more densely sampled since people tend to photograph them more making it more likely for the system to find an instance level match. In the original im2gps, a minor majority (58%) of matches were instance level matches. Still, a significant portion of images were successfully located even though the database lacked an exact match.

In the original im2gps paper scenes were matched using a combination of several types of features. These features include Tiny Images[11], color histograms, texton histograms, line features, Gist[9],

Figure 5: Example landscape versus non landscape images. The two images on the left are landscapes, while the two on the right are not.

and geometric context. Scale invariant feature transform (SIFT)[6] and Maximally stable extremal regions (MSER)[7] features were also added in later experiments in , which helped to improve performance significantly especially for instance level recognition. For matching images we use both the old and new features, giving us twenty two features per image and a feature dimension of about 40000.

Hays [3] examines feature performance for geolocation, showing that each feature contributes differently to geolocation accuracy, with color and Gist and texton features being relatively powerful by themselves, while Tiny Image and geometric context features having the worst relative performance. Combining all these features results in better performance across all the images in the test set, indicating that these features are latching on to different types of image information. This does not mean that all features should be treated equally for all images. While for all classes of images, using all features improves average performance, for specific types of images, we find that using all feature types actually can decrease performance. In other words, what features work well for one type of image may not work so well for another type of image.

We examine the specific case of landscape class images in relation to feature performance. Landscape images are among the hardest to geolocate since many similar appearing landscapes may occur in different parts of the world, but they make up a significant portion of the im2gps database. [5] To examine feature performance on landscape images, we extracted 10320 randomly chosen images from the im2gps database. These photos were then manually classified into landscape and non landscape images. Of these photos, 1029 were classified as landscape images, with the remaining 9291 photos being classified as non landscape (~11% of photos were landscape images). We tested four feature categories - color features, SIFT and MSER features, texture features, and line or GIST features, and experimented with different combinations of these feature categories.

Using all the features on our landscape only test set of 1029 images resulted in 67 correct 1NN matches within 200km (139 if within 750km). The best performing set of features was Gist, color, and texture features, but no SIFT or MSER features (72 within 200km and 142 within 750km), while the worst performing set of features was Gist and SIFT or MSER, but no color or texture features (34 within 200km and 70 within 750km). These results show that for landscape images, color and texture type features are more important than Gist or SIFT features. Since landscapes of

[5] Hays [3]shows that landscape geolocation is relatively bad for both humans and im2gps, with less than 10% of landscape images being correctly located to within 400km using the original im2gps features.

depict similar textures (ground, vegetation, sky, etc.) on varying terrain, this is not too surprising of a result.

It is important to note that the difference between using all features and just the best set of features is rather small, and we would need a larger test set to determine if there is a significant performance change between using those two sets of features. In contrast, using just the GIST and SIFT or MSER features yields much poorer performance than the two previously mentioned sets of features.

4 Incorporating Stronger Features

While im2gps uses a fairly exhaustive set of features, most of the current im2gps features rely on local gradient or color information, with very little knowledge about the scene itself. When humans are asked to geolocate images, they often latch onto higher level information such as objects, structure etc. We would like to incorporate higher level scene understanding into the pipeline with the hope that they would help improve performance on difficult to locate images.

We focus on adding information about scene structure, since im2gps has no knowledge about 3d scene shape. Objects with similar textures at different scales will have a strong match with each other under the current system. This can be troublesome when an image with trees in the distance is matched to another image with grass in the foreground (see Figure). We would like to incorporate some knowledge about the scene geometry into the system to help disambiguate these types of mistakes.

One way to encode scene geometry is via a depth map. Unfortunately, accurate depth from a single image is a difficult task, partially because this is inherently an under constrained problem of reprojecting 2D points back into 3D. .Make3D[10] demonstrates a method for estimating depth from a single image. Unfortunately their system was trained on range finder data with a maximum range of 81 meters, we wish to estimate depths far greater than that often seen in landscape imagery. To accomplish this, we first collect known depth data for landscape imagery.

Since we do not need exact estimates of depth, but rather approximate estimates, and since accurate depth information for large open spaces is difficult to collect, we instead use human annotation to give approximate depth information. Depth estimates are collected using Amazon's Mechanical Turk (AMT) by first automatically segmenting several randomly chosen images from the database. Gingold et. al [2] experimented with micro computation using turkers to estimate depth layers, however we require more information than just an ordering of segments.

We use VLFeat's[12] implementation of the Simple Linear Iterative Clustering (SLIC) superpixel segmentation algorithm[3]. SLIC superpixels were chosen to segment the image since they provide fairly uniform segments which follow the boundaries well, compared with other superpixel algorithms which may segment the image into non uniform regions. For our experiment we prefer uniform regions since it is possible to control region size which is important when estimating depth per region (large similar looking regions of an image may still cover a large depth extent). Example segmentations are shown in Figure 4.

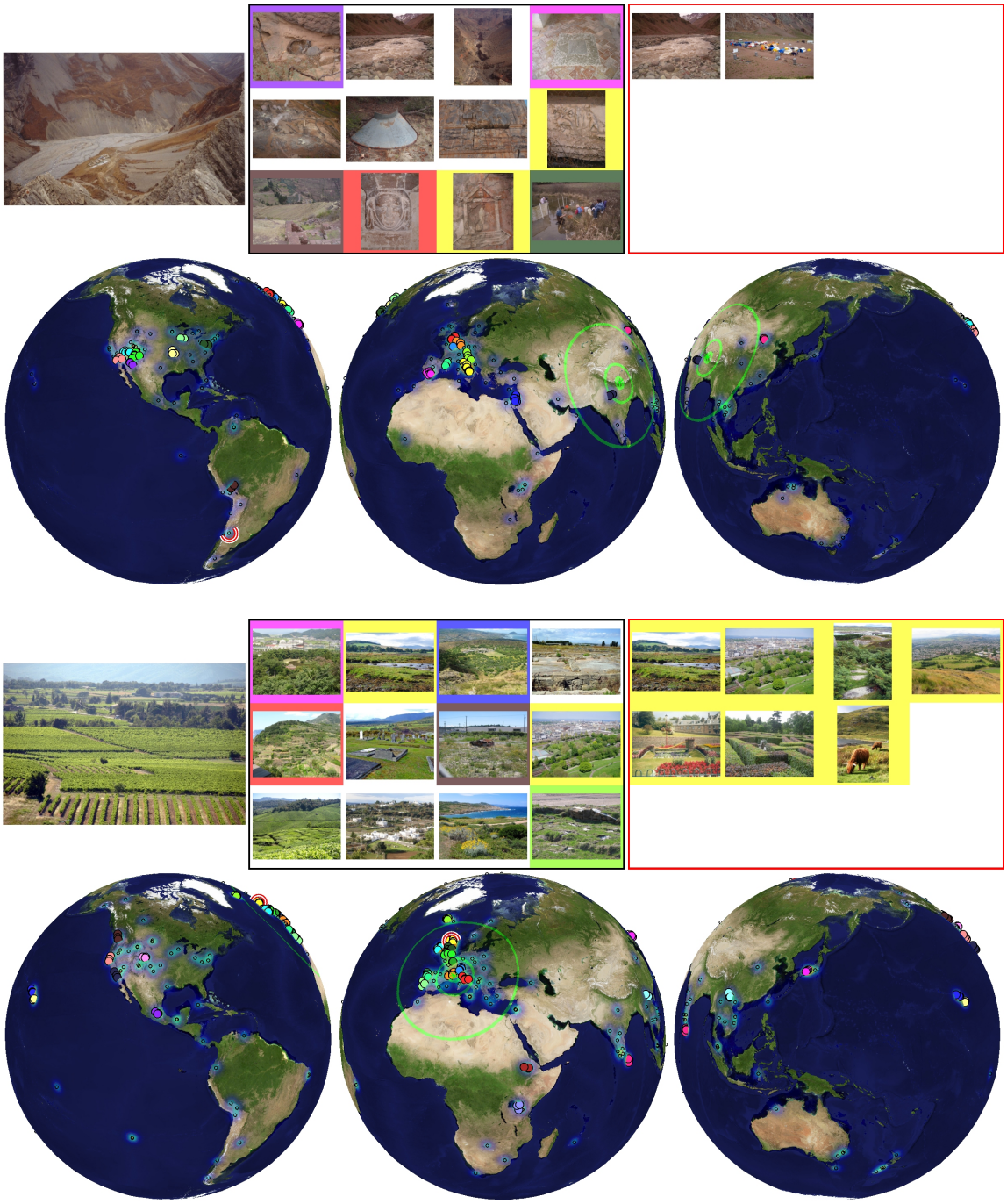

Figure 6: Some examples of incorrectly geolocated images using k=200 neighbors. The query photograph is displayed to the left, the top scene matches with their cluster color are displayed in the middle, and the top cluster after learning is displayed on the right column. The estimated location is given by the red-white target circle and the ground truth location is shown by the three concentric green rings (at distances of 200, 750, and 3000km). Note that the top scene matches are similar in scene layout and texture, but not in 3D scene structure.

Figure 7: Sample SLIC segmentations used in our AMT experiments. Segments are assigned different colors in each image. Each segmentation consists of about twenty different segments. Ideally, more segments would be used, but cost is a limiting factor.

4.1 Human Depth Estimation

With the hope of collecting data to use for estimating depth from single images, we first evaluate human performance on estimating absolute depth. Particularly, we want to know if humans can give reliable enough depth estimates to use for computation.

For each segment in each image, we ask AMT workers to estimate the distance from the camera. Each hit comprised of five randomly selected regions from randomly selected images. For each image, the worker is asked to enter a numerical answer for how far away from the camera they estimate that segment to be. If the segment is infinitely far away (for example, the sky), they are asked to select a check box indicating that the segment is infinitely far away.

This process was repeated 10 times over 26 different images, with each image segmented to an average of about 20 segments per image. Each image was resized to a maximum size of 640 pixels in either dimension before segmenting. [6]For each hit we paid \$0.05, or \$0.01 per image (since there were five images in a hit). While most results retrieved were good answers to question, there were some workers who input bad answers. To filter out these bad answers, the median of depth estimate is compared against each answer and answers above a certain threshold from median are thrown out.

We compared the mean, median and standard deviation of the depth estimate in each segment (see Figure 4.1).

[6]Using VLFeat's SLIC implementation on an image with region size of 130 and a regularizer value of 130 gave about 20 segments for each image.

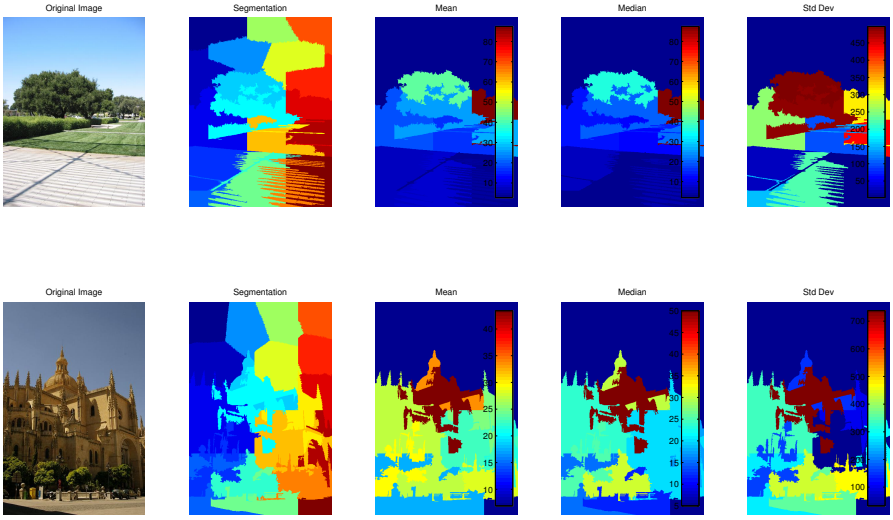

Figure 8: Example results from human depth estimates. To the left is the input image, which is segmented and then submitted to AMT. The results are aggregated and the mean, median, and variance are computed for each segment. Note that all units are meters and that infinite distances (ie. the sky show up as dark blue, in these images.

In general the depth estimates in most results are reasonable. Workers mostly agreed on the depth ordering, even if they disagreed on the exact depth somewhat. Most unreasonable results come from poor segmentations, which can be remedied by using more segmentations - increasing the number of segments to 50-60 per image at $0.03 for every 5 segments is feasible. As shown in the examples, standard deviation is usually higher in segments which are farther away from the camera, which is not surprising.

The depth estimates gathered from turkers are discontinuous and constant for each segment, which is not entirely desirable. Ideally depth maps would vary smoothly over regions of varying distance from the camera and then jump over discontinuous edges. The easy solution would be to increase the number of segments, but this becomes prohibitively expensive very quickly. To try and create smooth depth variation we create another AMT experiment. In this experiment we use the same images and segmentations as with the absolute depth experiment. Similar to the absolute depth experiment, the workers is presented with five images per hit, each with a highlighted region. For each region the worker is asked to input which direction of the segment is farthest away from the camera. We ran this experiment three times and paid $0.03 per hit. Given the direction of depth change, we can combine the absolute depth information at each segment to create a smoother depth map.

Unfortunately, as shown in in Figure 4.1, the results were rather noisy and not entirely correct when verified. Worker agreement was also somewhat poor. Whether this is a function of the question or

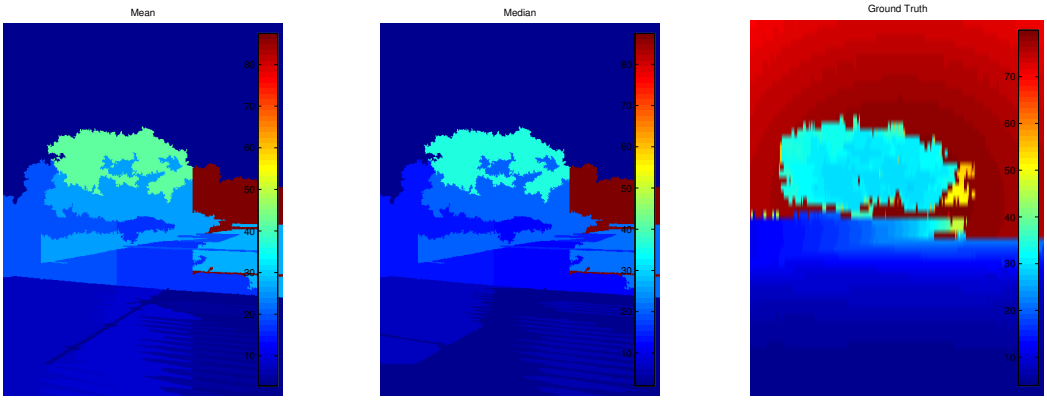

Figure 9: Our estimates (left two images, mean and median of segmentation values) versus the ground truth data from Make3D laser range scanner collection. Units are in meters. Again, segments which are infinitely far away show up as zero (blue) in our visualizations.

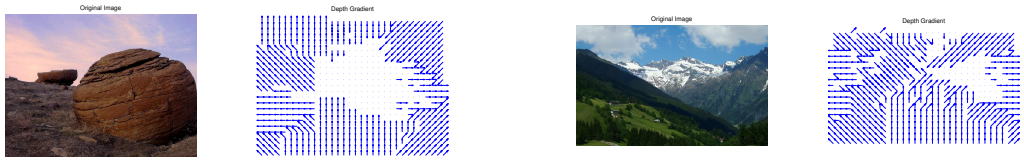

Figure 10: Depth directions collected from AMT visualized using MATLAB's quiver plot. For each segment, arrows are pointing towards the part of that segment that is further away from the camera. Most of the results were noisy and incorrect.

the relatively small number of repeat hits (3 versus 10) is not clear, however, getting more results per segment would probably help disambiguate worker uncertainty. It may also help to increase the number of segments to around 30-40 per image.

5 Future Work

We have started the collection of one of the largest geotagged image databases. While the database collection and preprocessing is still ongoing, once it has finished, we would expect to see an increase in performance in im2gps and other similar tasks which require matching similar scenes. Some possible applications which could directly benefit from the larger database might be data driven scene completion [5] , or super resolution [11].

While the database in quite large, it does consist of many "bad" photographs - photographs which do not help with the geolocation problem. These photos might be useful for other tasks, but for the purpose of geolocation, we would like to filter these photos. We are hesitant to run automatic classifiers to detect bad photos, to avoid biasing our data, since many of the features used in these classifiers are also used by im2gps. One option would be to manually try and remove bad photos using AMT, however given the large number of photos this might be somewhat costly. Furthermore, the criteria for determining which photos are good and bad for geolocation is not immediately obvious. Photos which may not be useful in the current system may become more useful in future versions of im2gps.

Unfortunately one of the downsides of working with large data is that most computations take much more CPU time to complete[7]. Due to time limitations we did not complete more experiments examining performance of the im2gps system incorporating the new data (both the new database, and depth estimation information). As the data scales up more and more it is also important to explore methods to speed up the search and computation especially for scene matching (the core part of many of these data driven algorithms).

In the future, it would also be interesting to see further experiments on feature performance for other classes of images besides landscapes. We believe that different classes of images will geolocate better using different sets of features. In the specific case of landscape images explored here, however, we believe that there is a lot more work to be done in terms of feature selection. Landscapes can often be located by looking at the type of ground, plants and the presence or absence of water. Features which can latch on to the subtleties of these variations (ie. tundra vs. snow), would certainly help in geolocating these images.

When trying to build depth estimates for images using Amazon's Mechanical Turk, we found that the resulting depth estimates were mostly good, but relatively coarse. On the other hand, there is a lot of work that can and should be done to improve the human estimated depth before trying to create a set of training images for learning depth information. Specifically methods to improve depth estimation and the quality of the depth without incurring too much additional cost should be explored.

Another possibility for improving geolocation performance using higher level scene understanding is to incorporate object detection into the pipeline. When asking humans to geolocate images, we found that they were often able to locate images accurately by recognizing text or certain culturally tied objects in the image. Being able to detect these objects and using them to help locate images would not be difficult to incorporate into the pipeline.

[7] Matching one image against the 6.5 million im2gps database takes around 15 minutes on a quad core machine. Matching against the larger database will probably take significantly longer. In our experiments, working with the larger 60 million image database presented its own set of problems - for example loading all the GPS data into MATLAB caused it to allocate over 12 GB of RAM before crashing. We alleviate this problem by chunking the data and and information.

6 Appendix

6.1 Storing and Using the Image Database

While we are storing a significant amount of data and files, we would like to maintain compatibility with the old im2gps, but avoid having tens of millions of images distributed over the file system. For this reason, all images are stored in a similar format to the original im2gps database. This includes a root directory containing folders with one folder per keyword (ie. America, Taipei). Within each of these location folders, the photos are stored in a series of numbered zip files, with each zip file containing ~1000 images (instead of a series of numbered folders like the old im2gps database). ZIP archives were chosen not for compression, but rather for grouping, and allow us to list and access files in the archive without extracting the whole archive. To work with these images, we have developed replacement tools for common MATLAB functionality (ie. imwrite, imread, imfinfo) which allows us to work directly with images stored in these archives (without needing to extract the whole zip file). These tools should ease im2gps migration to the new database (ie. imwrite_zip, imread_zip, imfinfo_zip). We plan to precompute and store features in a similar manner.

6.2 What Keywords Do Users Annotate Their Images With?

To help determine our choice of negative keywords we created a histogram of tags for all 6.35 million photographs in the original im2gps database and manually searched for keywords describing images which were not geographically relevant. The im2gps database consisted of 48902 unique tags. See Figure 11. Selecting the correct keywords to use is an import part of the data collection process since this directly determines how noisy the database is. While using both positive and negative keywords helps limit the noise of the database, there is still much room for improvement.

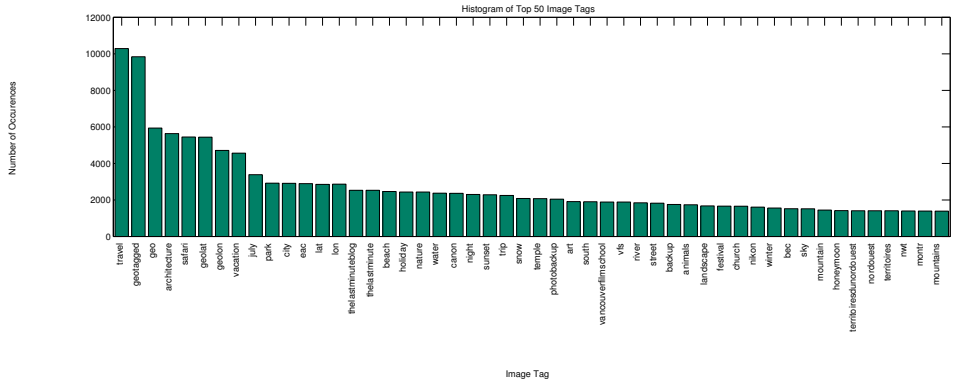

Figure 11: Histogram of the top 50 image tags excluding tags which are less than three letters long and positive keywords used to query images.

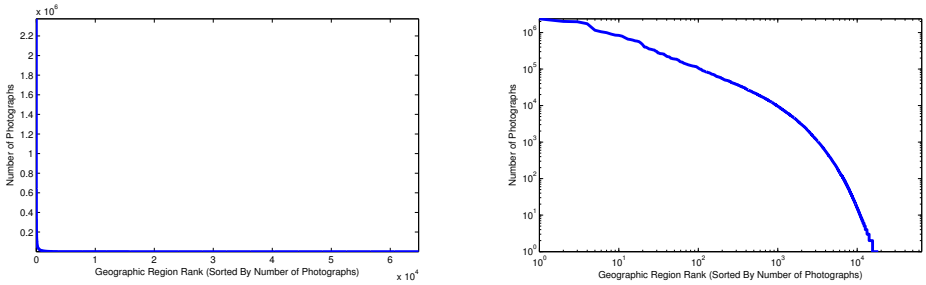

Figure 12: Plot of the geographic distribution of photos. The left plot is shown on a linear scale while the right plot is shown on a log scale.

Since the database includes tag information for each image after downloading, it is possible to use these tags for other purposes. For example, given these tags it would not be difficult to add image category labels to each image (from a source of possible labels such as WordNet [8]) to create a very large image category database.

6.3 Photo Density Rank in the New Database

As addressed in [3], a common argument against purely data driven methods is the tendency of data to follow heavy tailed distributions. Heavy tailed distributions are undesirable for data driven methods because there may not be enough samples at the tails to work with. Specifically, given too few images at certain locations in the world, data driven geolocation will not have a good chance of ever accurately matching to those photos. Similar to [3], we subdivide the earth into 64800 regions and compute a histogram of images (using their geo coordinates) over these regions. We then sort these regions by number of photographs, giving us a geographic region rank. As shown in Figure 6.3, the distribution appears to be heavy tailed in the left, but looking at the log-log plot on the right shows that the distribution of photos is not too heavy-tailed. Compared to the old im2gps database, however, the new database's distribution is slightly more heavy tailed.

7 Acknowledgements

We would like to thank James Hays for his role as an advisor for this project.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC Superpixels. EPFL Technical Report 149300, June 2010.

[2] Y. Gingold, A. Shamir, and D. Cohen-Or. 2011. Micro Perceptual Human Computation for Visual Tasks ACM Trans. Graph. V, I, Article A (Month 2011), N pages.

[3] J. Hays. Large Scale Scene Matching for Graphics and Vision. PhD Thesis, School of Computer Science, Carnegie Mellon University. July 2009.

[4] J. Hays and A. A. Efros, Im2gps: estimating geographic information from a single image. IEEE Conference on Computer Vision and Pattern Recognition. 2008.

[5] J. Hays, A. A. Efros. Scene Completion Using Millions of Photographs. ACM Transactions on Graphics (SIGGRAPH 2007). August 2007, vol. 26, No. 3.

[6] D.G. Lowe. Object recognition from local scale-invariant features. ICCV, pages 1150– 1157 vol.2, 1999.

[7] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. Proc. BMVC, 2002.

[8] G. A. Miller (1995). WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41.

[9] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. Int. J. Comput. Vision, 42(3):145–175, 2001.

[10] A. Saxena, S. H. Chung, A. Y. Ng. 3-D Depth Reconstruction from a Single Still Image. International Journal of Computer Vision (IJCV), Aug 2007.

[11] L. Sun, J. Hays. Super-resolution from Internet-scale Scene Matching. Proceedings of the IEEE Conf. on International Conference on Computational Photography (ICCP), 2012.

[11] A. Torralba, R. Fergus, and W. T. Freeman. Tiny images. Technical Report MIT-CSAIL-TR-2007-024, 2007.

[12] A. Vedaldi and B. Fulkerson, VLFeat: An Open and Portable Library of Computer Vision Algorithms, 2008, http://www.vlfeat.org/.