https://doi.org/10.2352/ISSN.2470-1173.2017.19.AVM-022

Motion Estimation Using Visual Odometry and Deep Learning Localization

Suvam Bag*, Vishwas Venkatachalapathy*, Raymond W. Ptucha

Abstract

Modern day vehicles and especially driver assisted cars rely heavily on advanced sensors for navigation, localization and obstacle detection. Two of the most important sensors are the Inertial Measurement Unit and the Global Positioning System devices. The former is subject to wheel slippage and rough terrain, while the latter can be noisy and dependent on good satellite signals. The addition of camera sensors enables the usage of visual data for navigation tasks such as lane tracking and obstacle avoidance, localization tasks such as motion and pose estimation, and for general mapping and path planning. The proposed approach in this paper allows camera systems to work in conjunction with or replace both Inertial Measurement Unit and the Global Positioning System sensors. The proposed visual odometry and deep learning localization algorithms improve navigation and localization capabilities over current state-of-the-art methods. These algorithms can be used directly in today's advanced driver assistance systems, and take us one step closer towards full autonomy.

Introduction

According to the Department of Transportation, US accidents in 2014 caused 32,000 fatalities and 2.31 million injuries. These accidents cost the US economy $1 trillion. Self-driving cars are predicted to reduce these accidents dramatically. Research at the University of Berkeley indicates autonomous cars will increase fuel efficiency by 21%, allow more cars to fit on the same road, and make more efficient use of limited parking space. Commuter times will be reduced, our air will be cleaner, and our quality of life will improve. While raw Inertial Measurement Unit (IMU) and Global Positioning System (GPS) sensory information is very helpful, it is not sufficient for self-driving cars. The Google smart car uses a complex path planning algorithm, whose heuristic function is dependent on LIDAR/camera information as well as GPS and IMU sensors. The proposed computer vision and deep learning approach can augment or even eliminate much of the need for GPS and IMU sensors.

One of the challenging tasks for smart cars is to track incremental motion in real-time and to analyze surroundings for accurate localization. This crucial information is used by internal systems such as active suspension control, autonomous steering, and lane change assist. Visual Odometry (VO) is the measurement of incremental change in pose or perspective using only visual information. VO provides incremental motion at shorter time intervals by tracking the motion of static background, but this solution is susceptible to drift over longer time intervals. To compensate for drift, typical systems re-position a vehicle at frequent intervals. Localization, the process of determining a vehicles position and pose on a static map, provides a global view of where the smart car is with respect to the environment.

Localization typically involves multiple sensors feeding complex algorithms. Even if a car has a high resolution static map of the world, perhaps from Google maps or other satellite generated maps, it isnt enough for navigating, as real world environments are dynamic. To adapt to these changes, a vehicle often creates a local map of its own and integrates this map with the global static map to identify its exact location. This local map is created by the various sensors present in the car such as a high dimensional LIDAR, a radar, and multiple cameras. After getting a sense of its local environment, the car can continue executing its navigation algorithm to reach its destination point. [1].

A* is a popular path planning algorithm used by autonomous vehicles. This algorithm has been improvised over the years with modifications such as dynamic planning, control parameters, stochastic solutions, and path smoothing. Localization and routing based on inaccurate GPS coordinates is difficult. The average root mean square GPS precision of a typical Google maps is 2-10 meters. This precision is sufficient for a human to drive and plan her next move but its not precise enough for a self-driving car. To achieve higher localization accuracy, smart vehicles utilize an ensemble of sensors. A typical smart car may use satellite maps for path planning, street view imagery for learning maps, and various sensors for lane marking, corner detection, and obstacle detection. To correct for sensor limitations and noise, various algorithms such as Monte Carlo and Kalman filters are used.

Related Work

Motion estimation using visual odometry and localization have typically been separate algorithms in smart cars, however recent success of efficient visual odometry algorithms and deep learning have motivated this research to consider merging the two algorithms into a single, unified approach. This paper presents an entirely vision based approach to provide a system with localization and path planning abilities. There have been many attempts in designing efficient algorithms for visual odometry and localization. While visual odometry has wide applications in advanced driver assistance systems (ADAS), localization is one of the basic requirements in path planning and simultaneous localization and mapping (SLAM) in the world of robotics.

Motion Estimation

Visual odometry is a concept derived from a problem commonly known as Structure From Motion (SFM) [33]. SFM is a problem of recovering relative camera pose and estimating its 3D structure from a set of cameras, which can either be calibrated or non-calibrated (epipolar plane). The concept of visual odometry was coined in 2004 in [13] , which used dense stereo matching along with optical flow to estimate motion. In [14] and [15] concepts related to 3D projections, camera calibration, and baseline optimization were introduced.

Harris and Pike [14] put forth the idea of position integration from consecutive frames to find out the end position with respect to the origin. SFM [27] covers a wide set of applications such as 3D reconstruction, but still needs visual odometry to track the position at which different image sets are taken. These image

sets may be consecutive or in random order, and hence are usually processed offline. The resultant structure and the pose of the cameras with which the images were captured are processed using offline optimization's like bundle adjustment [16]. Bundle adjustment can be used to refine the local estimate of the trajectory. While bundle adjustment [16] works on image sets that are captured non-consecutively, VO processes image sets taken sequentially to track incremental changes that help in building a resultant motion map.

Visual odometry needs to processes large sets of independent image frames in real time. In the early 1980s, Moravec [17] started solving the problem of a vehicle's egomotion from visual input alone. Much of the early research following Moravec [19] was aimed at precise visual odometry for planetary rovers and it gained more interest from NASA's Mar's exploration program. It was during this period when a lot of advantages and drawbacks of using a visual-only method for tracking a vehicle's egomotion were discovered. These outcomes inspired this paper's research into visual odometry.

The Mars rover's needs necessitated 6-degree-of-freedom (DoF) for rover's motion to overcome difficult issues such as wheel slippage in rough terrains. Lacroix et al. [18] described the importance of the key points in his implementation of stereo visual odometry for planetary exploration rovers. They used a dense stereo matching approach to cluster regions with similar depth and to track the motion of this region. The idea behind this approach was that the background could be classified into regions like buildings and trees and then tracking these regions would result in better accuracy. The dense features were selected by using a correlation function in the neighborhood of the pixel to cluster pixels into regions of similar depth.

Localization

Localization can be performed using various sensor platforms such as global positioning systems (GPS), LIDAR, monocular and stereo vision, omnivision, RADAR, wheel encoders, etc. The precise localization of self driving cars is not based on the feedback of one particular sensor but rather an intelligent combination of the output of multiple systems. Some of the techniques as well as historical methods will be discussed below.

One of the primary challenges in localization is to adjust the car's local position in a dynamic map with respect to a global map. One of the relatively simple but popular methods is known as Super Cruise. Super Cruise uses an array of sensors, lasers, RADAR, cameras, and GPS technology to analyze a car's surroundings. Super Cruise is a combination of two technologies. The first is the increasingly common adaptive cruise control, which uses a long-range radar (more than 100 meters) in the vehicle grille to keep the car a uniform distance behind another vehicle while maintaining a set speed. The second, lane-centering, uses multiple cameras with machine-vision software to detect road lines and objects. This information is sent to a computer that processes the data and adjusts the electrically assisted steering to keep the vehicle centered in the lane. [2] proposes a unique method of using the vision feedback in a map which has been learnt by LIDAR scans. This technique of using one of the sensors for lane tracking while using another for a much broader localization is how many modern autonomous vehicles perform localization and mapping.

[5] used traditional computer vision methods like Scale Invariant Features (SIFT) [31] and Random Sample Consensus (RANSAC) [28] as interest points for place recognition. [6] used a dataset of 100,000 GPS location tagged images from the Google Street View along with hand crafted vision features to classify and predict the location, from both a single query image as well as a group of neighboring images using a hierarchical model. In [22], the Monte Carlo localization method has been used in a Visual SLAM (Simultaneous Localization and Mapping) environment which has been an active area of research. VO has been determined to be an effective mode of localization and mapping in different topological conditions such as underwater using essentially the same basic principles [23]. In recent times, VO has reached new heights in localization as we have seen in the Mars Rover [24] and autonomous vehicles [25]. A broader view of mapping and localization of smart cars can be understood from [26].

In recent years deep learning has revolutionized the world of machine learning, pattern recognition, computer vision, and robotics. In many of these cases, it has been found that deep learning has been able to produce better detection ability than humans when given specific tasks. Convolution Neural Networks (CNNs) are feed-forward artificial neural networks where banks of filtered images are constructed in a hierarchical fashion, forming layers of increased abstraction. CNNs have proved to yield better results than traditional techniques which use hand-crafted features for detection [3]. In 2015, Lin et al. [4] presented a method that used CNNs for ground to areal geolocalization. The primary advantage in using CNNs in vision based localization is its ability to preprocess images through its layers. CNNs have proved to be more effective in matching images or identifying objects in recent years and may one day make traditional feature detection techniques obsolete. One of the major advantages of the CNN architecture is its co-learning of features with classifier, giving it an advantage over hand crafted features pared with independently learned classifiers. The disadvantage of CNN lies in its requirement of huge training datasets making it computationally expensive during training. However, despite arduous training time, processing test frames is extremely efficient making it suitable for real time applications. Deep learning frameworks have addressed the need for large training sets and lengthy training times by using transfer learning and utilizing parallel threads on Graphical Processing Units (GPUs) respectively.

Challenges

Lack of identification of proper objects

Despite using state-of-the-art object recognition algorithms, most autonomous cars lack the ability to identify what we may consider to be simple objects. For example they may fail to differentiate between pedestrians and police officers. It is difficult to differentiate between puddles and potholes, as well as recognizing people's actions such as a police signaling the car to stop. Despite being able to parallel park in controlled situations, it is still difficult to train a vehicle to go park itself in an arbitrary parking lot. Although these might seem relatively easy compared to what has already been achieved, they are actually difficult and require sophisticated machine learning.

Weather/Climate/Time of day

The capability of most sensors change drastically under different weather conditions and different times of day. For example

a LIDAR performs very well in clear conditions but its performance drops significantly in rainy or snowy conditions. A lot of research and improvement needs to be done on this front so that the autonomous cars become globally acceptable in every country and in every state, besides performing with the same precision throughout the day. Examples of synthetically added rain and snow on a selected image out of our datasets have been shown in Figure 1. Experiments are being conducted to use these images in an effort to make detection systems weather invariant.

Figure 1. *(a) Original image; (b) Synthetically added rain; and (c) Synthetically added snow.*

Loop closure detection in SLAM

One of the challenges associated with SLAM is to solve the loop closure problem using visual information in real-world situations. The term loop closure detection is primarily defined as the computers ability to detect whether it is in the same place or not after traveling a certain distance. The difficulty of this task is in the strong appearance changes that a place suffers due to dynamic elements, illumination, weather or seasons. This area of SLAM hasnt been perfected yet, but vision based localization plays a key role here. Previous approaches to this problem include FAB-MAP [7], which is a topological appearance based SLAM which has been regarded as one of the most reliable and stable approaches.

Since change in illumination and weather affects place recognition, [8] has proposed a different approach called Seq-SLAM to acknowledge this problem. Seq-SLAM removes the need of a global matching performance by calculating the best candidate matching location within every local navigation sequence instead of calculating the single location most likely given a current image. [9] reviews other methods to address scene recognition. Another very interesting problem is bidirectional loop closure detection [10], which tests the autonomous cars ability to detect its location irrespective of its direction of approach.

Method

Our proposed approach for Visual sensing for automobiles can be divided into three major parts motion estimation using visual odometry, localization using deep learning, and the combination of the two to make a robust system.

Motion estimation using visual odometry

A stereo camera setup was placed on top of an autonomous golf cart for data collection, as shown in Figure 4. The setup consists of two identical cameras attached to the roof 11.5 cm apart. The sensors lie on the same plane and small setup errors were corrected using stereo camera calibration. Figure 3 shows two different configurations that can be made in order to capture stereo images. The setup used in our approach is the parallel sensor configuration to cover a larger area of the environment and to focus on more objects. This setup was used for real-time ego-motion estimation and path correction using deep learning localization. The setup outputs 640x480 images with a 60 degree field of view at 18 fps.

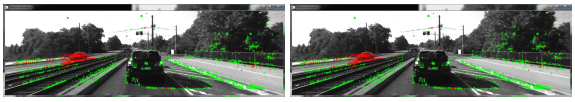

Figure 2. *Feature points from stereo images.*

The stereo image sets were rectified to satisfy epipolar geometry and the images were converted to gray scale for faster processing. Since the feature detection is only intensity level based, gray scale images provide sufficient information.

1. If the stereo image set is the first in its sequence, then the image is only used to generate a 3D feature set. The initial feature generation stage is also performed if the tracking information is lost. At this stage features from accelerated segment test are used with bucketing approach to generate a steady flow of features.
2. The image is first divided into segments by windowing the image.
3. Each window will have an initial Fast Threshold value, which will be adaptively updated based on the number of features generated in that window. Using the Adaptive Fast Threshold value, fast features are generated in each window separately.
4. These features are matched from the left image (L_t) to the right image (R_t) in the image stereo set to get feature correspondence and to generate the feature depth using *equations 1-4*. Their location is made precise by using sub pixel interpolation. With the features' location and depth information, it becomes a three dimensional feature set.

$$disparity = X_{left} - X_{right} \quad (1)$$

$$X_{realworld} = \frac{x - c_x}{disparity} * T \quad (2)$$

$$Y_{realworld} = \frac{y - c_y}{disparity} * T \quad (3)$$

$$Z = \frac{f}{p} * \frac{T}{disparity} \quad (4)$$

where,
$disparity$ = depth of features in Z direction.
C_x = X axis variation of the image plane
C_y = Y axis variation of the image plane
f = focal length of both the cameras
T = distance between the cameras
p = distance between pixels inside the camera sensors.

5. If the stereo image set is not the first in its sequence then,
6. The three dimensional features from the previous image set (L_{t-1} and R_{t-1}) are tracked to the current stereo sets, left image(L_t) using LucasKanade optical flow technique [30].
7. Follow step 1c to find feature correspondence between the left (L_t) and the right(R_t) images of the stereo image sets. Only features that are tracked from the previous results to the current frame are considered.
8. At this point two sets of three dimensional features corresponding to two consecutive frames have been obtained. Now the problem is to find the orientation and translational changes between the three dimensional feature set. At first

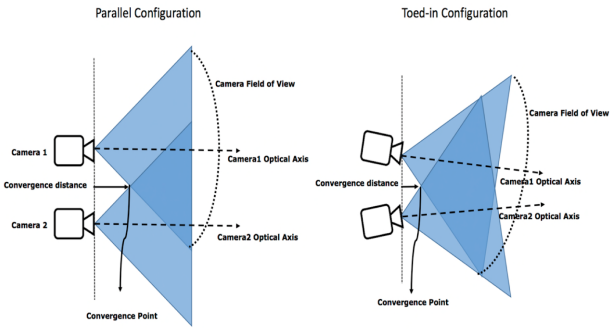

Figure 3. *Parallel vs Toed-in Configuration*

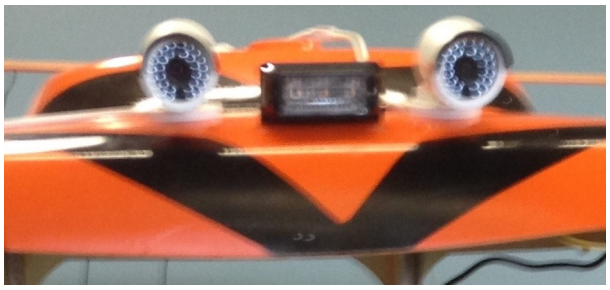

Figure 4. *Hikvision stereo camera setup*

we divide the three dimensional features into subsets and perform RANSAC [28] using Horns Method [29] to find out the weighted closed form solutions for absolute orientation. The band of results are considered to find the median pose.

9. After the Motion estimation step using Horns method [29], the features are weighted based on their contribution towards the final result.
10. The result is further corrected by using pose results of previous frames.
11. The features' variance in motion from $t-2$ to $t-1$ and $t-1$ to t frames is recorded and used to predict if a feature is a good feature or not. Features that are consistent in all the previous frames ($L_{t-4}L_{t-3}L_{t-2}L_{t-1}L_t$) are assumed to be robust and to provide more information. The features' predicted motion is a continuation of its motion from the previous frames. The variance from its predicted motion to the actual motion being tracked in the current frame is used to weigh features. A weight of 0, is assigned to features that have huge variances in motion and such features are removed later.

Bucketing is the term used for dividing the image into multiple ROIs, each of which is used for generating interest points. This approach assumes each segment of the image is a completely independent image and generates FAST features [32] with a score depicting the strength of the interest point. The strength of the interest points are computed using the difference in the intensities between the surround pixels and the center pixel. This approach is very fast and effective as it can be easily parallelized with features generated across the image. Using bucketing with adaptive FAST thresholds help to a maintain robust and steady feature count. The thresholds are adjusted adaptively to acquire a similar number of features from each of the segments of the image. By adjusting the FAST threshold for each segment independently, features from different contrast backgrounds can be generated. For example, in a bright sunny day, the features on a tree or on a building wall facing the opposite direction of the sunlight would be masked by the sunlight. Using Adaptive FAST thresholding with bucketing, a robust set of features from both the bright and dim regions can be generated.

The motion estimation approach provides inter frame pose estimation and correction with robust FAST features [32]. The use of adaptive featuring helps in generating features across different regions of images with varying brightness, and helps detect robust features. The process of using a pyramidal approach aids in creating better depth resolution for each feature and hence precise motion along the Z axis. Feature weighting helps in removing outliers which induce more error in the estimation process. Over time, the small errors for each pose prediction accumulates and drifts away from the actual position and hence a correction mechanism is required. Use of deep learning to predict the global position helps to recover the drift as described in the next section.

Localization using deep learning

Outdoor localization based on vision is a challenging problem but due to the recent advancements in deep learning, these challenges are being overcome. Given a sufficient set of imagery, deep learning models are shown to be quite good at predicting precise location. Moreover, the learned models can predict the location in real time. Transfer learning and data augmentation can also be used to make these datasets more robust. Our approach to this problem builds a dataset with classes corresponding to unique pairs of latitudes and longitudes. By augmenting the images for different fields of view, heading and pitch, we obtain the same location images from different viewpoints. The following approaches have been used to create datasets:

- Google Street View - Datasets were created from the Google Street View images of the university campus, where each class corresponds to a bunch of images from different viewpoints at a unique pair of latitude and longitude. The unique waypoints (green and red markers) were selected as landmark locations in the campus (Figure 5).

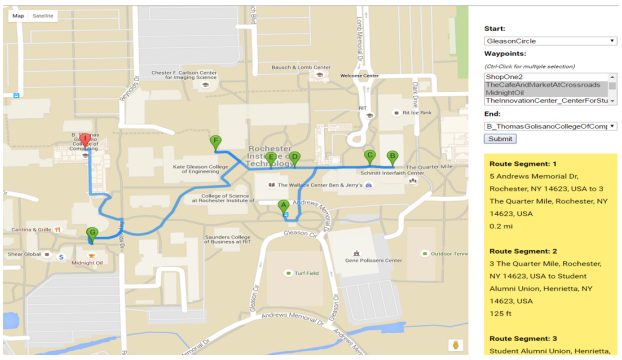

Figure 5. *Paths between waypoints.*

Using the Google Maps API, a set of coordinates were obtained from the connected paths between these waypoints in a way such that the viewpoints were non-redundant

(Figure 6). Building the dataset in this format helped in eliminating redundant images for close latitudes and longitudes. A few examples have been shown in Figure 7

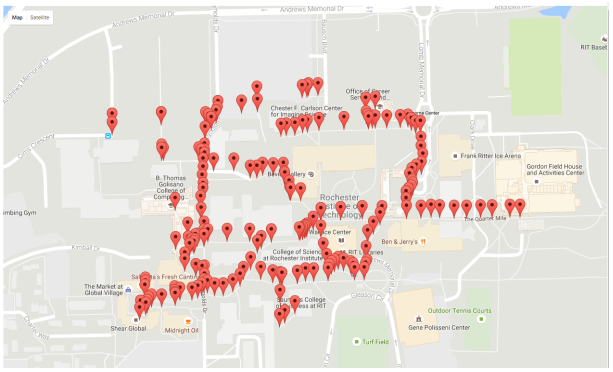

Figure 6. *Classes from the Google Maps API.*

Figure 7. *Google Street View images from different classes.*

- Dataset created manually - Following the map created out of Figure 6 as closely as possible and covering as many similar points, a dataset was created by driving an autonomous golf cart (Figure 8) around the campus. A pair of Hikvision Bullet cameras were used for acquiring the images Figure 4.

Figure 8. *Autonomous golf cart.*

- Classifier - Inspired by the recent success of CNNs in classification, the popular AlexNet [11] was used for classifying the datasets. AlexNet was the one of the first revolutionary deep learning architectures which successfully classified ImageNet [12] with a top-1 accuracy of 57% and a top-5 accuracy of 80.3%. The original architecture had five convolutional layers connected with two fully-connected layers in the end. The architecture used in this paper is a modified version of the original AlexNet architecture with batch normalization included between the convolutional layers. One single block of this architecture consists of *convolutional layer* $\rightarrow$ *batch normalization* $\rightarrow$ *ReLU activation layer* $\rightarrow$ *Max Pooling layer*. The code-base for the classifier was used from the popular imagenet-multiGPU code developed by Mr. Soumith Chintala under the Torch framework. [34]

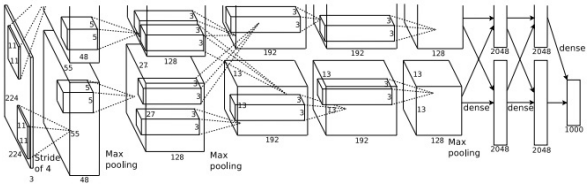

Figure 9. *Original AlexNet architecture [11].*

Motion estimation combined with localization

CNN deep learning models are trained on manually collected images, Google Street View images as well as manually collected images augmented with Google Street View. Each image corresponds to a unique pair of latitude and longitude positions. During test time, given an unforeseen image, the CNN estimates the initial vehicle position in the world followed by incremental motion being estimated over time by VO. To combat error propagation produced by VO, CNN localization is continuously interleaved with VO motion estimation. The CNN prediction ensures the vehicle stays on track, nullifying the possibility of a large pose estimation error. The architecture has been shown in (Figure 10). The value of n was selected to be '10' in this paper i.e. - the motion estimation module would receive the ground truth after processing every $10th$ frame in a sequence from the localization module trained on the same set of images.

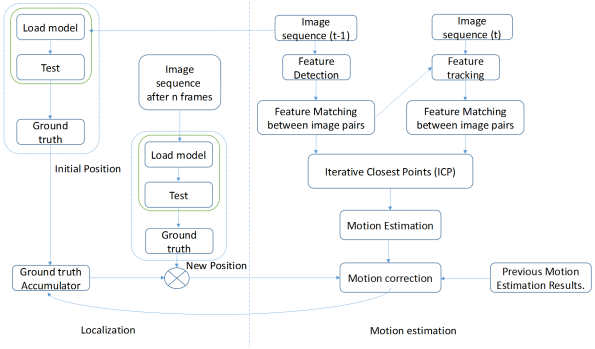

Figure 10. *Flow chart of the entire process.*

Results

There were broadly two different datasets created for experimentation. They are described below -

Google Street View Dataset

Using the Google Maps API, 26 important waypoints were selected out of the entire university campus. Permutations of route segments were taken between these waypoints. All the GPS coordinates were collected from each of these route segments. This dataset was rich, because many of the unnecessary GPS coordinates were neglected. The final points primarily consisted of coordinates from the paths inside campus, where a golf cart would be able to travel (Figure 6). A similar technique can also be applied outside campus for normal cars on public roadways.

- Total number of images - 36K approximately
  Total number of classes - 166
  Parameters -
  1) Field of View (range 0, 120)
  2) Heading (range 0, 360)
  3) Pitch (range -90, 90)
  Images per class - 216
  Model - AlexNet with batch normalization
  Best validation accuracy - 75.4%

Dataset from the golf cart

The golf cart dataset was created by driving the golf cart around the university campus. The ground truth was obtained from the network using a MacBook pro which gets the GPS coordinate using the laptops location. The different precision of the GPS sensors unfortunately meant the ground truth differed in this dataset compared to the ones created out of Google Street View. This highlights the importance of sensor robustness, whereas a vision based system would be much more robust. In both the datasets, the GPS coordinates were acquired with up to six floating point precision wherever available. Ideally a six floating point corresponds to approximately 2-10 cm precision. However since this is not the root mean square precision, such accurate precision is not available throughout, and our proposed method would help the system. Some examples have been shown in Figure 11. The ground truth vs. predicted class are shown in Figure 12. A total of 714 points are shown in the map. The correct predicted points are marked as red, and the incorrect ones as green. It can be observed here, that even the few incorrectly predicted points are close to the ground truth.

- Total number of images - 7120 approximately
  Total number of classes - 89
  Images per class - Not constant
  Model - AlexNet with batch normalization
  Best validation accuracy - 97.2%

Cross testing

The dataset collected driving the golf cart around was tested using the model learned from the Google Street View images. The ground truth vs. predicted classes are shown in Figure 13. It should be noted here that although the red markers are quite distinct to the green markers, it is not necessarily a proof of incorrect

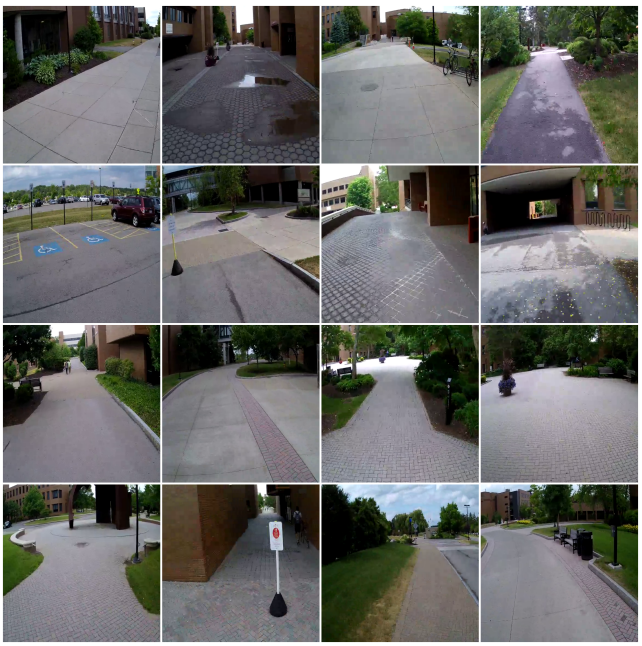

Figure 11. *Dataset created using the golf cart.*

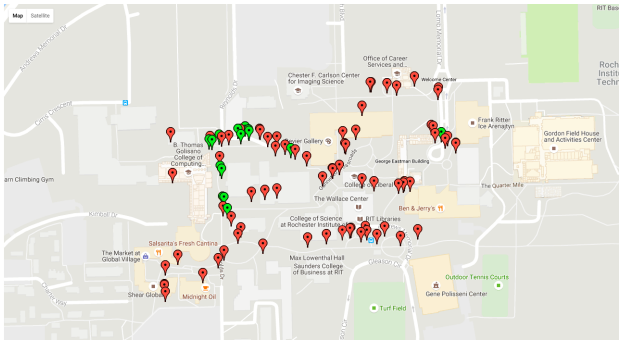

Figure 12. *Correct vs incorrect prediction - dataset from the golf cart.*

predictions as the ground truth (labels) are different in the Google Street view dataset compared to the ground truth (labels) of the golf cart dataset. The difference is distinct because of the difference in precision in the GPS sensors used in collecting data. The VO model can be compared with both the predictions.

Motion estimation with localization on dataset from the golf cart

An accurate measurement of global position using a very precise GPS can be expensive and still suffers measurement inaccuracies due to weather and satellite communication. In contrast, a predictive learned model based on a single time measured global position is very appealing due to cost, invariance to weather, and ability to drive close to or in the shadows of large buildings. Other approaches for post processing path corrections are either complex or not real-time. The proposed approach of using deep learning to know the global location, and to correct any deviations from the commute path can be computed in real-time and is more precise for a well learned model.

In our approach, the local motion estimation provides dis-

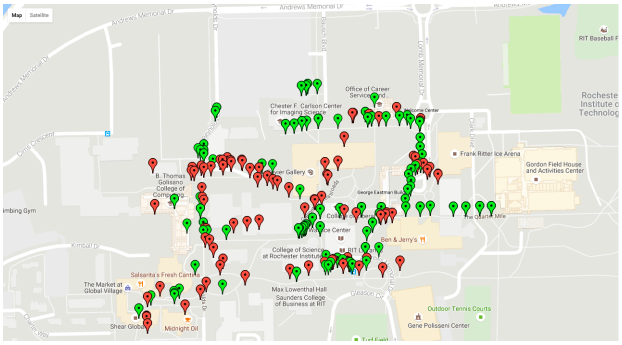

Figure 13. *ground truth of Google Street View vs predicted class from cross testing.*

placement in D_x and D_y (Displacement along X and Y axis respectively), which is later converted to GPS coordinates by updating it with the inertial global positioning results from the deep learning model. At regular intervals in time, the deviation produced by the motion estimation model is rectified by taking the average of the results from the same model and the predictive model. In Figure 14, the red markers show the ground truth GPS coordinates, the green markers show the incorrect predicted coordinates from the deep learning localization model, and the blue markers show the incorrect results from the predictive model aided by the motion estimation algorithm.

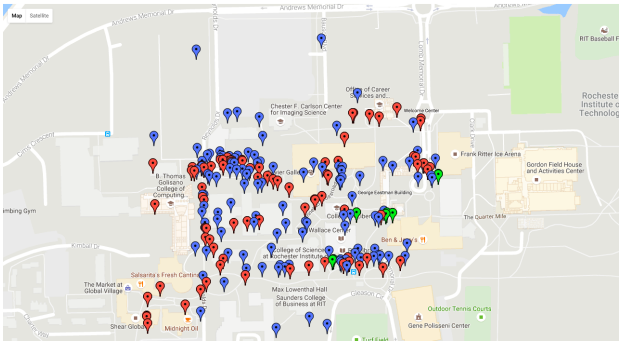

Figure 14. *Motion estimation aided by the predictive model.*

Conclusions and Future work

This paper presents a motion estimation and localization approach that specifically solves propagation of error from VO-only approaches. Our deep learning predictive model pairs computer vision with precise GPS values to localize a smart vehicle on a large university campus with an accuracy of more than 97%. It has been shown in Figure 12 that if a localization model is well trained, then the motion estimation model can rectify its deviation based on the values from the predictive model alone, instead of having to average multiple models values (blue markers in Figure 14.

Future work includes recurrent deep methods for predicting global position. These methods will be especially useful when the motion estimation results change very drastically from its previous poses. More experiments can be conducted on better training the network with only structural images and not planar images where a plain field or a wall is present as they don't provide any information for classifying. Regression based models will be used for predicting exact global location and motion estimation. Other approaches for improvements include monocular motion estimation and localization using models trained on street view images. This approach did not perform well in this research due to the difference in the precision of the GPS sensors used in Google maps and our experiments. But given similar GPS sensors, we believe that the proposed method may work quite well. A typical challenge in localization is to adapt the model to the dynamically changing environment due to change in time of day or weather/climate. A lot of research has been conducted to solve this problem [20] [21]. It was observed while conducting experiments, that localization is most precise when models are trained in a hierarchical fashion on the datasets i.e - smaller models are trained on different sections of the dataset. This idea can be applied in improving the precision of traditional GPS sensors as well.

References

[1] S. Thrun ; J.J. Leonard; "Simultaneous Localization and Mapping," in Springer *Handbook of Robotics*, vol 23, no. 7-8, 2008, pp. 871-889

[2] R. W. Wolcott and R. M. Eustice, "Visual Localization within LIDAR maps for Automated Urban Driving," *IEEE Int. Conf. Intell. Robot Syst.*,pp. 176-183,2014.

[3] Z. Chen, O. Lam, A. Jacobson, and M. Milford, "Convolutional Neural Network-based Place Recognition," 2013.

[4] T. Lin, J. Hays, and C. Tech, "Learning Deep Representations for Ground-to-Aerial Geolocalization," pp. 5007-5015, 2015.

[5] M. Yu and U.D. Manu, "Stanford Navigation Android Phone Navigation System based on the SIFT image recognition algorithm,"

[6] A. R. Zamir and M. Shah, "Accurate Image Localization Based on Google Maps Street View," Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 255268. Springer, Heidelberg (2010)

[7] M. Cummins and P. Newman, "FAB-MAP: Appearance-Based Place Recognition and Mapping using a Learned Visual Vocabulary Model," *Proc. 27th Int. Conf. Mach Learn*, pp. 3-10, 2010.

[8] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2012, pp. 1643-1649.

[9] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, J. J. Yebes, and S. Bronte, "Fast and effective visual place recognition using binary codes and disparity information," *2014 IEEE/RSJ Int. Conf. Intell. Robot Syst,* no. SEPTEMBER, pp. 3089-3094, 2014.

[10] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, J. J. Yebes, and S. Gamez, "Bidirectional loop closure detection on panoramas for visual navigation," *IEEE Intell. Veh Symp. Proc.,* no. JUNE, pp. 1378-1383, 2014.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Adv. Neural Inf. Process. Syst., pp. 19, 2012.

[12] J. D. J. Deng, W. D. W. Dong, R. Socher, L.-J. L. L.-J. Li, K. L. K. Li, and L. F.-F. L. Fei-Fei, ImageNet: A large-scale

hierarchical image database, 2009 IEEE Conf. Comput. Vis. Pattern Recognit., pp. 29, 2009.
[13] H. Longuet-Higgins, A computer algorithm for reconstructing a scene from two projections, Nature, vol. 293, no. 10, pp. 133135, 1981.
[14] C. Harris and J. Pike, 3d positional integration from image sequences, in Proc. Alvey Vision Conf., 1988, pp. 8790.
[15] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu,Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys, Building Rome on a cloudless day, in Proc. European Conf. Computer Vision, 2010, pp. 368381.
[16] B. Triggs; P. McLauchlan; R. Hartley; A. Fitzgibbon (1999). "Bundle Adjustment A Modern Synthesis". ICCV '99: Proceedings of the International Workshop on Vision Algorithms. Springer-Verlag. pp.298372.
[17] L. Matthies and S. Shafer, Error modeling in stereo navigation, IEEEJ. Robot. Automat., vol. 3, no. 3, pp. 239248, 1987.
[18] Olson, L. Matthies, M. Schoppers, and M. W. Maimone, Robust stereo ego-motion for long distance navigation, in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2000, pp. 453458.
[19] C. Harris and J. PikeH. Moravec, Obstacle avoidance and navigation in the real world by aseeing robot rover, Ph.D. dissertation, Stanford Univ., Stanford, CA, 1980.
[20] X. Fu, J. Huang, X.Ding, Y. Liao and J. Paisley, "Clearing the Skies: A deep network architecture for single-image rain removal," arXiv:1609.02087v1 [cs.CV] 7 Sep 2016
[21] Y. Li, R. T. Tan, X. Guo, J. Lu and M. S. Brown, "Rain Streak Removal Using Layer Priors," CVPR 2016
[22] H. Lategahn, A. Geiger, and B. Kitt, Visual SLAM for autonomous ground vehicles, Proc. - IEEE Int. Conf. Robot. Autom., pp. 17321737, 2011.
[23] S. S. D. C. Botelho, P. Drews, G. L. Oliveira, and M. D. S. Figueiredo, Visual odometry and mapping for underwater autonomous vehicles, 2009 6th Lat. Am. Robot. Symp. LARS 2009, 2009.
[24] M. Maimone, Y. Cheng, and L. Matthies, Two years of visual odometry on the Mars Exploration Rovers, J. F. Robot., vol. 24, no. 3, pp. 169186, 2007.
[25] I. P. Alonso, D. F. Llorca, M. Gavilan, S. A. Pardo, M. A. Garcia-Garrido, L. Vlacic, and M. A. Sotelo, Accurate Global Localization Using Visual Odometry and Digital Maps on Urban Environments, Ieee Trans. Intell. Transp. Syst., vol. 13, no. 4, pp. 15351545, 2012.
[26] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, Towards fully autonomous driving: Systems and algorithms, IEEE Intell. Veh. Symp. Proc., pp. 163168, 2011.
[27] K. Hming & G. Peters (2010). "The structure-from-motion reconstruction pipeline a survey with focus on short image sequences". Kybernetika.
[28] M. A. Fischler & R. C. Bolles (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". Commun. ACM. 24: 381395.
[29] B.K.P. Horn and B.G. Schunck, "Determining optical flow." Artificial Intelligence, vol 17, pp 185203, 1981. Manuscript available on MIT server.
[30] B. D. Lucas and T. Kanade (1981), An iterative image registration technique with an application to stereo vision. Proceedings of Imaging Understanding Workshop, pages 121–130.
[31] Lowe, David G. (1999). "Object recognition from local scale-invariant features". Proceedings of the International Conference on Computer Vision. 2. pp. 11501157.
[32] Edward Rosten and Tom Drummond, Machine learning for high speed corner detection in 9th European Conference on Computer Vision, vol. 1, 2006, pp. 430443.
[33] F. Dellaert; S. Seitz; C. Thorpe & S. Thrun (2000). "Structure from Motion without Correspondence". IEEE Computer Society Conference on Computer Vision and Pattern Recognition.
[34] S. Chintala, "https://github.com/soumith/imagenet-multiGPU.torch", Copyright (c) 2016, Soumith Chintala All rights reserved

Author Biography

Suvam Bag is pursuing his MS in Computer Engineering from the Rochester Institute if Technology, with a special focus on machine learning and self-driving cars. He is doing his master's on finding a novel method of deep learning localization for self-driving cars.

Vishwas Venkatachalapathy completed his MS in Computer Engineering from the Rochester Institute of Technology. with a deep focus on computer vision and machine learning. Since his graduation, he has been working as a deep leaning software developer at the Valeo Corporation in Santa Carla, USA.

Raymond Ptucha is an Assistant Professor in Computer Engineering and Director of the Machine Intelligence Laboratory at Rochester Institute of Technology. His research specializes in machine learning, computer vision, and robotics. Ray was a research scientist with Eastman Kodak Company where he worked on computational imaging algorithms and was awarded 30 U.S. patents with another 19 applications on file. He graduated from SUNY/Buffalo with a B.S. in Computer Science and a B.S. in Electrical Engineering. He earned a M.S. in Image Science from RIT. He earned a Ph.D. in Computer Science from RIT in 2013. Ray was awarded an NSF Graduate Research Fellowship in 2010 and his Ph.D. research earned the 2014 Best RIT Doctoral Dissertation Award. Ray is a passionate supporter of STEM education and is an active member of his local IEEE chapter and FIRST robotics organizations.