Visual Localization Under Appearance Change: A Filtering Approach

Anh-Dzung Doan, Yasir Latif, Thanh-Toan Do, Yu Liu, Shin-Fang Ch’ng, Tat-Jun Chin and Ian Reid

Abstract—A major focus of current research on place recognition is visual localization for autonomous driving, which must be robust against significant appearance change. This work makes three contributions towards solving visual localization under appearance change: i) We present G2D, a software that enables capturing videos from Grand Theft Auto V, a popular role playing game set in an expansive virtual city. The target users of our software are robotic vision researchers who wish to collect hyper-realistic computer-generated imagery of a city from the street level, under controlled 6 DoF camera poses and different environmental conditions; ii) Using G2D, we construct a synthetic dataset simulating a realistic setting, i.e., multiple vehicles traversing through a road network in an urban area under different environmental conditions; iii) Based on image retrieval using local features and an encoding technique, a novel Monte Carlo localization algorithm is proposed. The experimental results show that our proposed method achieves better results than state-of-the-art approaches for the task on visual localization under significant appearance change. The dataset will be available online upon acceptance. G2D is made available at: https://github.com/dadung/G2D

I. INTRODUCTION

To carry out higher level tasks such as planning and navigation, a robot needs to maintain, at all times, an accurate estimate of its position and orientation with respect to the environment. When the robot uses an existing map to infer its 6 degree of freedom (DoF) pose, the problem is termed localization. In the case when the map information is appearance (images) associated with different parts of the map, the problem is that of visual localization (VL). Image based localization methods normally assume that the appearance remains unchanged from when the map is generated to the present time when the robot needs to localize itself. However, as the operational time span of the robot increases, the appearance of the environment inevitably changes. This poses a great challenge for visual localization methods as the underlying assumption of static appearance is violated due to either changes in seasons, weather conditions, and times of the day or a combination of all these. One approach towards dealing with appearance change is to observe as many variations as possible of each location and carry a representation that can model them [1]. However, the bottleneck in this case is the significant amount of time needed to capture sufficient naturally occurring appearance variations. For example, recent benchmarks [2], [3], [4] have taken years to collect data with sufficient natural appearance variation. An orthogonal approach to address appearance change, pioneered in SeqSLAM [5], is to consider sequence-to-sequence matching instead of matching a single query image to previously observed images. SeqSLAM showed that sequence based matching is more resilient to appearance change but comes with its own shortcoming (e.g., sensitivity to view-point changes and differences in sequence velocities [6]). While recent learning-based VL algorithms [7], [8], [9], [10] have focused on the case of a single query image, in a robotics setting, the camera is operating continuously once the vehicle is moving and hence, it is realistic to expect a video (sequence of images) as a input to VL methods.

This work addresses the problem of metric visual localization (that is, we recover the 6 DoF pose of the camera) under appearance change, in the context of autonomous vehicles. To facilitate observation of different condition, we introduce a data generation tool named G2D which is used to create a benchmark dataset. Inside a Monte Carlo localization scheme that can reason along the temporal dimension (sequences), we propose a novel image encoding mechanism that generates image representations that are robust to appearance change. More specifically, our contributions are as follows:

1) To facilitate data collection with appearance variation, we present "GTA to Data (G2D)" - an image simulator tool that exploits the detailed virtual environment in Grand Theft Auto V (GTA V). It allows users to collect hyper-realistic computer-generated imagery of urban scenes, under controlled 6DOF camera poses and varying environmental conditions (weather, time of day, traffic density, etc.).

2) Using G2D, we generate a synthetic dataset simulating multiple vehicles in different time of day and various weather conditions. Fig. 1 illustrates the setting of our synthetic dataset. Note that there is an increasing recognition of the value of synthetic dataset towards building autonomous driving systems [11].

3) To effectively reason over image sequences, we use a Monte Carlo algorithm that approximates the probability distribution of 6-DoF camera poses incrementally. We show that for the case of driving on urban streets, a simple motion model is enough to successfully track the vehicle’s pose. For the observation model, we propose a novel observation encoder based on image retrieval which generates a fixed (low) dimensional vector representation for each image. We show experimentally that such representation is more resilient to appearance change along with minor changes in the viewpoint, while being compact in size.
We experimentally validate the proposed localization method on our benchmark dataset as well as on the real dataset [4] to show that it outperforms state-of-the-art approaches in the task of 6-DoF visual localization.

II. RELATED WORKS

A. Synthetic data generation tools

Several existing works use virtual worlds for generating image datasets. CARLA [12] provides a virtual world for autonomous driving systems. SYNTHIA [13] and UnrealCV [14] create synthetic images for various computer vision tasks, based on their own virtual worlds.

Another paradigm for generating simulated images is to exploit readily made virtual worlds inside computer games. A primary target has been GTA V, which is set in a hyper-realistic urban environment. Krähenbühl [15] intercepts the DirectX rendering pipeline of GTA V to obtain the ground truth labels for semantic segmentation, depth estimation, optical flow, intrinsic image decomposition and instance tracking. Richter et al. [16], [17] inject a middleware between the game and graphics hardware to collect graphical information, based on which, they can construct ground truth for several computer vision problems, including visual odometry. The camera poses for visual odometry are recovered from the recorded meshes and their transformation matrices.

Different from Krähenbühl [15] and Richter et al. [16], [17], G2D has direct access to the native functions of GTA V for data capture, enabling us to obtain accurate camera poses in the games’ frame of reference for every frame of the game. Additionally, users can control the environmental conditions through functions inside G2D enabling collection of imagery under different conditions and the trajectory collection are highly automated.

B. Datasets for autonomous driving

In the recent work of Sattler et al. [2] which is aimed at autonomous driving, several state-of-the-art algorithms were compared on three datasets: Aachen Day-Night, RobotCar Seasons and CMU Seasons. Although the datasets include substantial environmental variations, and the testing images are sequential in RobotCar Seasons and CMU Seasons, the testing regime does not examine the benefit of temporal continuity for VL. Although there is a section in [2] on “multi-image queries”, it is a relatively small part of the benchmark, with only two methods tested; SeqSLAM [5] and an extension of Active Search [18] that uses triplets of images. The former is known to suffer significantly from slight pose variation [6], while the latter does not fully take into account temporal continuity. It is worth mentioning that [2] suggests “multi-image localization” as a promising future research.

Another recent dataset for VL is ApolloScape [3], which was collected in various weather conditions, but it only covers 7 roads in 4 different cities, and the video sequences were recorded in daytime. In addition, their evaluation metric, which is the median errors of translation and rotation, does not completely reflect the accuracy of VL in the scenario of autonomous driving. In our experiment, we observe that
even when the median errors between different methods are comparable, the smoothness of the predicted poses can be different (See section [V] for more details).

Apart from VL, there are many datasets aimed towards tasks related to autonomous driving. CamVid [19] is presented for semantic segmentation and camera pose estimation, but it is relatively small with 4 video sequences in total duration of 20 minutes. KITTI benchmark suite [20] provides ground truth for various computer vision tasks (e.g., stereo reconstruction, optical flow, visual odometry, object detection and tracking). Cityscapes [21] and Mapillary Vistas [22] target the semantic segmentation problem.

Our dataset is challenging as the training and testing sequences are significantly different in terms of their perceptual appearance (see Fig. 1) and has been created specifically to test the robustness of VL algorithms against significant changes in environmental conditions.

C. VL algorithms

Visual localization is a very well-studied area and various methods have been proposed that address the problem in different ways. For the scope of our work, we limit the survey to methods that can perform metric localization, that is, they recover the 6-DoF pose of the camera based on image information. These algorithms can be categorized into local feature-based methods and learning-based methods.

Broadly speaking, local feature-based methods estimate the pose of the query image from 2D-3D correspondences by solving the PnP problem [23] or an equivalent step [2]. The main difficulty for this approach is establishing and validating the 2D-3D correspondences for large (city-scale) maps. Many methods attempt to improve this step by using geometric constraints [7], [24], semantic information in the images [25], and learning-based matching [9].

Learning-based methods that perform 6-DoF pose estimation as an image-to-SE(3) regression problem. PoseNet [8] uses a convolutional neural network (CNN) to learn the regression mapping. LSTM-Pose [26] argues that the usage of a fully connected layer in PoseNet possibly leads to the overfitting, despite the usage of dropout. Therefore, they use Long-Short Term Memory (LSTM) units. Other variants of PoseNet include: uncertainty in the pose prediction [27] and geometric constraints [10], [28].

Our proposed method is different from the above methods, in that we seek not only the current 6 DoF pose of the camera as a point estimate but as a distribution over the space of all possible locations in the map. Therefore, we employ a probabilistic filtering framework, namely Monte Carlo localization, that takes into account the constraints on motion of the vehicle and the temporal nature of the localization task. To deal with appearance change, we propose a novel observation encoder based on image retrieval which relies on dense local features and encoding technique to ensure the robustness against environmental changes. While, previous works [29] [30] [31] have combined Monte Carlo localization with image retrieval for localizing a robot in an indoor environment, they only consider the case of a robot moving on plane (3-DoF) while we aim to recover the complete 6-DoF pose of the robot. Also, while these works [29] [30] [31] only test their method in the indoor environment, in which the appearance changes are insignificant; we show that due to the reduced size of the representation, our method can perform robustly in a large-scale outdoor environment with significant appearance changes, i.e., the car traverses over 10 km in its run.

III. Collecting data from GTA V

In this section we provide a brief overview of our tool G2D. Further technical details about G2D can be found in our technical report [32]. G2D is publicly available at https://github.com/dadung/G2D.

A. Methodology

For visual localization, we want to record two things from GTA V: images and their corresponding ground truth pose. To make the task of image collection easier and as automatic as possible, two types of camera trajectories are used in G2D: sparse and dense trajectories. The sparse trajectory consists of a set of user-define vertices (positions on the “top down” 2D map of the virtual environment), along with the order of visitation. The protagonist in the game automatically traverses the user-defined sparse trajectory, during which the 6-DoF pose of the camera is recorded at 60 frames per second leading to a dense trajectory. This reduces user effort and enhances repeatability. Finally images are collected by retracing a continuous path along specified by the dense trajectory. Each image captures the scene as observed from the gameplay camera. This step is also performed automatically.

G2D also allows the user to manipulate in-game conditions such as weather, time of day, and traffic density. This manipulation is done before the image collection phase, which allows the user to capture images under different conditions from the exact same set of 6-DoF ground truth poses as specified by the dense trajectory.

B. Synthetic dataset

Using G2D’s accurate pose and image collection mechanism, we construct a large-scale map for the task of visual localization. The collecting process is shown in Fig. 3. We simulated 59 vehicles running independently along different routes, weather conditions and time of day. The birds eye view of the dataset is shown in Fig. 2 with a coverage area of about 3.36km². We also simulated different times of the day and weather conditions, whose distribution is shown in Fig. 2. The times and weather conditions in the training sequences are uniformly distributed from 1am to 11pm, and in 7 different weather conditions (snowy, foggy, clear, overcast, cloudy, sunny and rainy). Five sequences at different times and weather were also collected for testing.

The dataset collection method ensures that the training and testing sequences vary in environmental conditions to reflect the scenario that we are interested in, i.e., continual

1Based on Intel i7-6700 @ 3.40GHz, RAM 16GB, NVIDIA GeForce GTX 1080 Ti and the highest graphical configuration for GTA V.
visual changes in the environment over a period of time. For the experiments presented in this work, we sub-sampled the training sequences at 2 Hz and the testing sequences at 4 Hz. Samples from training and testing sequences are shown in Fig. 2 and Fig. 3 respectively.

Now that we have the dataset in place, we present our localization method in the next section.

IV. MONTE CARLO-BASED VISUAL LOCALIZATION

Let the 6-DoF camera pose be given by: \( s_t = [r_t, \Omega_t]^T \), where \( r_t \) and \( \Omega_t \) represent the 3D position and Euler orientation respectively at time \( t \). The idea of Monte Carlo based visual localization is to represent the probability distribution \( p(s_t | u_{1:t}, z_{1:t}) \) by \( N \) particles, where \( u_{1:t} \) and \( z_{1:t} \) are motion and observation inputs up to time \( t \). Particularly, a set of particles are maintained at time \( t = \{ s_{t}^{[1]}, s_{t}^{[2]}, ..., s_{t}^{[N]} \} \) with their corresponding weights: \( \{ w_{t}^{[1]}, w_{t}^{[2]}, ..., w_{t}^{[N]} \} \). The states of particles are updated according to a motion model, and their weights are updated based on the observation encoder.

In the following sections, we justify the use of a simple motion model and present details of the novel observation encoder.

A. Motion model

When navigating the roads of an urban area, the motion of the car is fairly restricted, i.e., it largely stays on a road network on an approximately 2D plane [34]. While the road surface still allows significant scope for movement (cf. Fig. 1), relative to the size of the map, the motion of the car is rather limited. This is echoed by the recent work of [2], who observed that there is “lower variation in viewpoints as the car follows the same road”. Hence, a Monte Carlo scheme with a simple motion model suffices to track the 6-DoF rigid motion of the car. In many cities, the road networks are complex Euclidean graphs. In fact, it is well known that using (visual) odometry alone, it is possible to accurately track a car on a 2D road map [35][36]. More fundamentally, this suggests that temporal continuity in the testing sequence (which is fully exploited by our method) strongly benefits VL.

Mathematically, for each particle, we model the noisy action consisting of the velocity \( v_{i}^{[t]} \sim N(\mu_v, \Sigma_v) \) and angular velocity \( \psi_{i}^{[t]} \sim N(\mu_\psi, \Sigma_\psi) \), where \( \mu_v \) and \( \mu_\psi \) respectively represent the linear and angular velocities. The accelerations are modeled by the noise covariance matrices \( \Sigma_v \) and \( \Sigma_\psi \).

For each particle, their motion in each time step is given by:

\[
u_{i}^{[t]} = [v_{i}^{[t]}, \psi_{i}^{[t]}]^T.\]

In practice, the \( \mu_v, \mu_\psi, \Sigma_v, \) and \( \Sigma_\psi \) can be either manually tuned, or estimated from training data [36].

While 3D positions can be easily updated by using simple additions, we convert two Euler angles to the Direction Cosine Matrix (DCM) [37], multiply two matrices and convert to DCM and multiply two matrices and convert to Euler angles. More fundamentally, the car is a non-holonomic system [33].

In more “localized” operations such as parking, where highly accurate 6-DoF estimation is required, it is probably better to rely on the INS.

More fundamentally, the car is non-holonomic system [33].

On uneven or hilly roads, accelerometers can be used to estimate the vertical motion, hence, VL can focus on map-scale navigation.

The method of [35] will give ambiguous results on non-informative trajectories, e.g., largely straight routes. Hence, VL is still crucial.
in an image sequence. In case there is mismatch between the actual motion and the one predicted by the motion model, such as during emergency maneuvers, the discrepancy would be reflected in the enlarged covariance estimate and resolved once motion returns to within normal bounds.

B. Observation encoder based on image retrieval

To make the image representation robust to appearance change, we seek for every image \( I \) a nonlinear function \( \tau(I) \) that maps \( I \) to a vector in a fixed dimensional space. To do so, we first densely compute SIFT features: \( \{x_i \in \)
\( \mathbb{R}^d | i = 1, \ldots, M \) over the image, which are then normalized as follows: i) \( L1 \) normalize every SIFT vector \( x_i = x_i / \|x_i\|_1 \), ii) square root each element \( x_i = \sqrt{x_i} \). This step is called RootSIFT normalization [38] and makes the Euclidean distance calculation among SIFT features equivalent to computing the Hellinger kernel.

Subsequently, we employ the embedding function VLAD [39] to embed SIFT features into a higher dimensional vector space. In particular, given a vocabulary learned by K-means: \( C = \{ c_k \in \mathbb{R}^d | i = 1, \ldots, K \} \), every SIFT feature is embedded as follows: \( \phi_{\text{VLAD}}(x_i) = [..., 0, x_i - c_j, 0, ...] \in \mathbb{R}^d \), where \( c_j \) is the nearest visual word to \( x_i \), and \( D = K \times d \). Note that different from Bag of Word (BoW), which embeds the feature vector as follows: \( \phi_{\text{BoW}}(x_i) = [..., 0, 1, 0, ...] \in \mathbb{R}^K \), where only \( j^{th} \) component of \( \phi_{\text{BoW}}(x_i) \) non-zero means that the nearest neighbor of feature \( x_i \) is visual word \( c_j \); VLAD considers the residual between a visual word and its nearest feature. Do et al. [40] show that VLAD is a simplified version of local coordinate coding [41], which tries to approximate a nonlinear classification function by a linear function.

From a set of embedded vector: \( \{ \phi(x_i) \in \mathbb{R}^D | i = 1, \ldots, M \} \), we aggregate them by the sum pooling to obtain a single vector representing the image \( I \): \( \tau(I) = \sum_{i=1}^n \phi(x_i) \). In the literature, there are several other ways for this aggregation step [42], [43], but for simplification, we choose sum pooling and show that it can perform good in practice.

One drawback of using local features in image retrieval is that the background (e.g., trees, sky, road, etc) features significantly outnumber features from informative objects (e.g., buildings). To alleviate this, we apply PCA projection, whitening and L2-normalization [44], which limits the impact of background features in the vector \( \tau(I) \).

During the testing phase, we calculate the similarity between the query and database images using L2 distance and retrieve top \( R \) database images with smallest distances. Next, mean-shift algorithm is applied on \( R \) retrieved images over the translational part of their poses. We then select the largest cluster, and calculate the mean of translation and rotation [45], which is viewed as a noisy measurement \( z \) from the image query \( I \).

C. Updating particle weights and Resampling

For every particle, its weight is computed as the following: \( w_i[t] = p \left( z_{[t]} | x_{[t]} \right) \propto e^{-\frac{1}{2}(z_{[t]}-w_{[t]}^{i})^T \Sigma^{-1}(z_{[t]}-w_{[t]}^{i})} \), where \( \Sigma \) is a covariance matrix which describes the noise of the measurement obtained by observation encoder. Then, all particle weights are normalized to ensure their summation equal to 1: \( \forall i, w_i[t] = \frac{w_i[t]}{\sum_{j=1}^{N} w_j[t]} \).

Finally, we resample particles based on their weights by stochastic universal sampling [46]. This resampling step prevents the degeneracy problem, which can occur in the long-term localization scenario. Fig. 6 shows the filtering performed by our proposed method. At the first iteration, hypotheses are randomly generated. Hypotheses with small weights vanish if they are inconsistent with the noisy measurement from the observation encoder. Finally, the ambiguity is resolved, and the particles successfully track the vehicle. It is worth noting that in the example shown, the query and retrieved images are from different times and weather conditions.

V. Experiments

In this section we present experiments to show that performance of the propose VL method on synthetic as well as real datasets. For quantitative results we report the mean and median translation and orientation errors. The translation error is calculated as the Euclidean distance \( ||x_{est} - x_{gt}|| \). The orientation error \( \alpha \) is computed as the angular difference \( 2 \cos(\alpha) = \text{trace}(R_{est}^T R_{gt}) \) between estimated and ground truth camera rotation matrices \( R_{est} \) and \( R_{gt} \).

A. Implementation details

In the observation encoder, we extract SIFT feature at 4 different scales with region width of 16, 24, 32 and 40, over a grid with spacing of 2 pixels, visual vocabulary size is \( K = 128 \). The SIFT features are embedded and aggregated using VLAD to obtain a single vector of length 16384, which is then projected to a 4096 dimensional space via PCA, whitened and L2-normalized. For nearest neighbors search, we set \( R = 20 \).

Particles are initialized from Gaussian distribution with the mean from the noisy measurement in the first frame. The covariance matrices for initializing 3D location \( t_{[0]}^{i} \) and orientation \( w_{[0]}^{i} \) respectively are \( \text{diag}(10,10,10) \) and \( \text{diag}(0.0001,0.0001,1) \). The parameters for our method are set as the following: \( \Sigma_0 = \text{diag}(5,5,5,0.0001,0.0001,0.0001) \), \( \Sigma_v = \text{diag}(1,1,0.01) \), \( \Sigma_w = \text{diag}(0.00001;0.00001;0.01) \), \( \mu_v = [0.1,0.1,0.01]^T \), \( \mu_w = [0.001,0.00001,0.01]^T \), where \( \text{diag} \) is a function that outputs a square diagonal matrix with the elements of input vector on the main diagonal. The number of particles is fixed to 1000. These are the default parameters for all experiments, unless otherwise noted.

B. Synthetic dataset

As a first experiment, we compare the performance of MapNet [28] against the propose observation encoder based on image retrieval. For our method, we use the output of the observation encoder as the final output and no filtering is done. This is to show the robustness of the observation encoder against appearance change. For this experiment, no temporal information is used.

As can be seen from Table II and Fig. 7 MapNet struggles to produce a reasonable result. This is because MapNet formulates the problem as an image to pose regression, whose underlying assumption of constant appearance is violated when the appearance of the environment changes. Moreover, repeated structures such as trees, sky, and road surfaces can leads to ambiguities for MapNet. In contrast, our observation encoder based on image retrieval applies state-of-the-art normalization techniques to reduce the negative impact from repetitive objects. Hence, image retrieval significantly outperforms MapNet. However, as Fig. 7 shows, using only image retrieval leads to non-smooth trajectory
estimates, hinting at the need for temporal reasoning. The results for the complete system, which employs a motion model to exploit temporal information, leads to a dramatic improvement of our method over image retrieval in terms of mean errors. The median errors are comparable between image retrieval and our method, showing that our method has a tighter error distribution.

C. Oxford RobotCar

In this section, we compare our proposed method to state-of-the-art approaches, i.e., PoseNet [8], MapNet and MapNet+PGO [28]. In particular, PoseNet directly regresses 6-DoF camera pose from an input image. MapNet receives videos as training data, hence its loss function minimizes absolute pose per image as well as the relative pose between consecutive images (temporal consistency), it is then followed by a fine-tuning step on unlabeled data with their visual odometry (VO). MapNet+PGO, in the inference step, fuses the prediction of MapNet with VO by using pose graph optimization (PGO) to ensure the temporal smoothness.

We follow the configuration suggested by [28]. The split of training and testing sequences are summarized in Table III. The experiment is conducted on the alternate and full routes with the length of 1 km and 10 km respectively. The training and testing sets are recorded in different time, sequences are in different weathers and times of day. Finally, a synthetic dataset is generated, in which all training and testing sequences are in different environmental conditions, and image sequences with their accurate 6 DoF ground truth poses. Using G2D, a synthetic dataset in GTA V. Due to the capability of assessing to other approaches.

As shown in Table IV, our method outperforms PoseNet, MapNet and MapNet+PGO with a large margin in the full route, i.e., 10 km running distance. The predicted trajectory comparison between our method and MapNet and MapNet+PGO is shown in Fig. [8]. Our method produces a smoother trajectory than MapNet and MapNet+PGO. In alternate route, although MapNet+PGO achieves the best quantitative result, our method outputs a smoothest trajectory, compared to other approaches.

VI. CONCLUSION

In summary, this paper presents G2D, an open-source software that assists robotic vision researchers in collecting dataset in GTA V. Due to the capability of assessing to the native functions, G2D allows users to control various environmental conditions, and image sequences with their accurate 6 DoF ground truth poses. Using G2D, a synthetic dataset is generated, in which all training and testing sequences are in different weathers and times of day. Finally, a novel Monte Carlo localization based on image retrieval is presented. The experimental results show that our method outperforms state-of-the-art algorithms in visual localization for autonomous driving.

REFERENCES

[1] W. Churchill and P. Newman, “Experience-based navigation for long-term localisation,” The International Journal of Robotics Research, 2013.

[2] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, et al., “Benchmarking 6DOF outdoor visual localization in changing conditions,” in CVPR, 2018.

[3] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, “The apolloscape dataset for autonomous driving,” arXiv: 1803.06184, 2018.

TABLE II: Mean (top) and Median (bottom) rotation and translation errors on our synthetic dataset.

| Route          | Purpose | Recorded   |
|---------------|---------|------------|
| Alternate      | Training| 26/6/2014, 9:23:58 |
| route (1 km)   | Unlabeled| 14/5/2014, 13:50:20 |
| Alternate      | Training| 26/6/2014, 8:53:56 |
| route (10 km)  | Unlabeled| 14/5/2014, 13:46:12 |
| Full route     | Training| 23/6/2014, 15:41:25 |
| Alternate      | Query   | 12/12/2014, 10:45:15 |
| route (1 km)   |         | 28/11/2014, 12:07:13 |
| Alternate      | Query   | 02/12/2014, 15:30:08 |
| route (10 km)  |         | 12/14/2014, 10:45:15 |
| Full route     | Query   | 09/12/2014, 13:21:02 |

TABLE III: The split of training and testing sequences in Oxford RobotCar dataset.

| Route          | Purpose | Recorded   |
|---------------|---------|------------|
| Alternate      | Training| 26/6/2014, 9:23:58 |
| route (1 km)   | Unlabeled| 14/5/2014, 13:50:20 |
| Alternate      | Training| 26/6/2014, 8:53:56 |
| route (10 km)  | Unlabeled| 14/5/2014, 13:46:12 |
| Full route     | Training| 23/6/2014, 15:41:25 |
| Alternate      | Query   | 12/12/2014, 10:45:15 |
| route (1 km)   |         | 28/11/2014, 12:07:13 |
| Alternate      | Query   | 02/12/2014, 15:30:08 |
| route (10 km)  |         | 12/14/2014, 10:45:15 |
| Full route     | Query   | 09/12/2014, 13:21:02 |

TABLE IV: Mean (top) and Median (bottom) rotation and translation errors on the Oxford RobotCar dataset.

| Route          | Purpose | Recorded   |
|---------------|---------|------------|
| PoseNet [8]    | 10.45m, 4.45° | 28.16m, 8.50° |
| MapNet [28]    | 6.02m, 2.38°  | 27.75m, 8.47° |
| MapNet+PGO [28]| 5.76m, 2.27°  | 5.06m, 2.48°  |
| Our method     | 7.15m, 3.89°  | 6.99m, 1.02°  |
|               | 6.74m, 1.70°  | 4.23m, 1.40°  |

TABLE V: The split of training and testing sequences in Oxford RobotCar dataset.

| Route          | Purpose | Recorded   |
|---------------|---------|------------|
| Alternate      | Training| 26/6/2014, 9:23:58 |
| route (1 km)   | Unlabeled| 14/5/2014, 13:50:20 |
| Alternate      | Training| 26/6/2014, 8:53:56 |
| route (10 km)  | Unlabeled| 14/5/2014, 13:46:12 |
| Full route     | Training| 23/6/2014, 15:41:25 |
| Alternate      | Query   | 12/12/2014, 10:45:15 |
| route (1 km)   |         | 28/11/2014, 12:07:13 |
| Alternate      | Query   | 02/12/2014, 15:30:08 |
| route (10 km)  |         | 12/14/2014, 10:45:15 |
| Full route     | Query   | 09/12/2014, 13:21:02 |

VI. CONCLUSION

In summary, this paper presents G2D, an open-source software that assists robotic vision researchers in collecting dataset in GTA V. Due to the capability of assessing to the native functions, G2D allows users to control various environmental conditions, and image sequences with their accurate 6 DoF ground truth poses. Using G2D, a synthetic dataset is generated, in which all training and testing sequences are in different weathers and times of day. Finally, a novel Monte Carlo localization based on image retrieval is presented. The experimental results show that our method outperforms state-of-the-art algorithms in visual localization for autonomous driving.

REFERENCES

[1] W. Churchill and P. Newman, “Experience-based navigation for long-term localisation,” The International Journal of Robotics Research, 2013.

[2] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, et al., “Benchmarking 6DOF outdoor visual localization in changing conditions,” in CVPR, 2018.

[3] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, “The apolloscape dataset for autonomous driving,” arXiv: 1803.06184, 2018.
Fig. 8: Results on alternate (top) and full (bottom) routes in Oxford RobotCar dataset. The length of alternate and full routes are 1 km and 10 km respectively. The green lines are ground truth, and red lines are predicted trajectories.
[43] N. Murray and F. Perronnin, “Generalized max pooling,” in CVPR, 2014.
[44] H. Jégou and O. Chum, “Negative evidences and co-occurrences in image retrieval: The benefit of pca and whitening,” in ECCV, 2012.
[45] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman, “Averaging quaternions,” Journal of Guidance, Control, and Dynamics, 2007.
[46] D. Whitley, “A genetic algorithm tutorial,” Statistics and computing, 1994.