Vehicle Trajectories from Unlabeled Data Through Iterative Plane Registration

Federico Becattini(✉), Lorenzo Seidenari, Lorenzo Berlincioni, Leonardo Galteri, and Alberto Del Bimbo

Media Integration and Communication Center (MICC), University of Florence, Florence, Italy
{federico.becattini,lorenzo.seidenari,lorenzo.berlincioni, leonardo.galteri,alberto.delbimbo}@unifi.it

Abstract. One of the most complex aspects of autonomous driving concerns understanding the surrounding environment. In particular, the interest falls on detecting which agents are populating it and how they are moving. The capacity to predict how these may act in the near future would allow an autonomous vehicle to safely plan its trajectory, minimizing the risks for itself and others. In this work we propose an automatic trajectory annotation method exploiting an Iterative Plane Registration algorithm based on homographies and semantic segmentations. The output of our technique is a set of holistic trajectories (past-present-future) paired with a single image context, useful to train a predictive model.

Keywords: Autonomous driving · Trajectory prediction

1 Introduction

Autonomous driving the past years has been one of the fields in which machine learning and artificial intelligence were applied the most. Even though significant steps forward have been made [2], the problem is yet far to be solved. The complexity stems from the many facets of different nature that need to be taken into account: in addition to the actual movement of the car itself, a thorough understanding of the surrounding scene needs to be obtained, both for what concerns static components such as road layout and other moving agents [3]. To allow an effective planning of a safe route towards its destination, the autonomous car needs to recognize other agents and model their dynamics to the point of predicting their future behavior.

Predicting agents' future trajectories is a problem that can benefit from a complete understanding of the scene. The surrounding layout acts indeed as a physical constraint that outlines the possible routes that the vehicle can undertake. Without relying on maps or geolocalization sensors though, scene comprehension based only on computer vision systems can turn out to be extremely complex due to occlusion, background clutter and scene variability. Scene parsing and semantic segmentation methods [6] can aid with this problem by providing a semantic category for each observed pixel.

On the other hand, modeling object dynamics from an autonomous car perspective is a hard task by itself. Since the observer is constantly moving, the first obstacle one has to deal with is separating the two observed motions: the real motion of agents and the apparent motion caused by the moving camera. The common approach in generating datasets to train autonomous vehicles involves the use of costly laser based range finders in order to obtain precise environment measurements and the integration of GPS sensors in order to refer such coordinates into the real world [11]. Currently dash cameras can be deployed at a very low cost on vehicles, indeed a simple video search for *Dash Camera* on a video repository such as Youtube yields hundred of thousands distinct results. Interestingly, mining videos from the web allows to obtain data on dangerous situations such as accidents which are not ethically reproducible in a controlled dataset.

In this paper we move the first steps towards a method that will allow to generate trajectory datasets from real-world scenarios without the need of an instrumented vehicle and hours of driving. We propose an automatic pipeline finalized to the generation of holistic trajectories composed by past-present-future positions of all other agents. We obtain trajectories for each frame in a video sequence, starting only from an RGB stream, without relying on complex sensors such as LIDARs or external sources like maps. Our pipeline is composed by several modules aimed at tracking both agents and the ground plane on which they are moving. By combining semantic segmentations and local descriptors we estimate a transformation to map the ground plane from one frame to another, enabling the projection of object positions through time, onto a desired frame (Fig. 1). We refer to this process as *Iterative Plane Registration (IPR).*

The paper is organized as follows. In Sect. 2 we frame our method into an appropriate literature review. Section 3 is dedicated to our proposed technique,

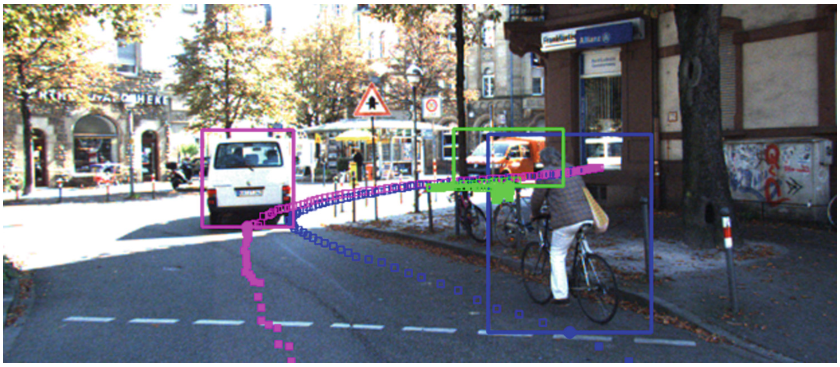

Fig. 1. Holistic trajectories shown on the reference frame. Past: full squares. Present: full circle. Future: empty squares.

providing an outline of the Iterative Plane Registration algorithm. In Sect. 4 we show the obtained results and we draw conclusions in Sect. 5.

2 Related Work

Recently several works targeted trajectory prediction [1,15,21]. The majority of this line of research targets non motorized vehicles and pedestrian trajectories [1,21]. For proper path planning of autonomous vehicles a full understanding of every moving agent behavior is necessary.

Collecting data for autonomous driving is a complex, slow and expensive procedure. Most autonomous driving datasets [7,11,14,18,25] are collected with cars equipped with several dedicated sensors: dash cameras provide footage, stereo rigs are used to obtain depth, laser scanners (LIDARs) generate cloud points, Inertial Measurement Units (IMU) log how the vehicle is moving and position is pinned down with GPS. As an example KITTI [11], provides all the above sources at 10 Hz.

The lack of trajectory information at a large scale is currently a limitation of many commonly used datasets, such as Cityscapes [7]. Only a few datasets nowadays contain trajectory information. KITTI [11] has a small fraction of the dataset annotated for object tracking; Berkeley Deep Drive (BDD) [25] provides instance level segmentations with consistent IDs across frames and nuScenes [5] has trajectory informations for the short video snippets that compose the dataset. None of these datasets offers a satisfactory number of trajectories to train a prediction model. The ApolloScape dataset [14] has been recently extended with approximately 80k trajectories for a new trajectory prediction task [18]. Trajectories are obtained combining LIDAR and IMU readings and are represented in a world reference system, which is the most common setting for this task [15,22]. Similarly, other common datasets dedicated only to pedestrian trajectories [20,21] are in a top view reference system. This way of representing data is easy to process and evaluate, yet is hard to obtain due to the need of a laser scanner and loosens the correlation between pixels and vehicle dynamics. Nonetheless, these datasets have a high cost. They require the instrumentation of a car with cameras, inertial sensors, gps and even more expensive sensors such as LIDARs. Moreover, it must be taken into account the human effort in driving the instrumented vehicle and in the annotation phase if no automatic object labeling and tracking is used.

Differently from previous approaches, we avoid these problems by collecting full trajectories directly in the frame reference, pairing past and future paths to what the car has in front, mimicking what humans see when driving. Furthermore we do not require any specific equipment and we work solely with RGB frames. This aspect also thins the acquisition process since any dash cam recorded video (even scraped from the web, e.g. YouTube) can be used to generate trajectories, instead of relying on heavily equipped fleets.

Simultaneous Localization and Mapping (SLAM) [4] is a basic tool for any autonomous driving platform, providing ego-motion estimation, 3D reconstruction and self-localisation in a single optimization framework. Recently deep learning based frameworks [23,24] have been used to improve classical feature based SLAM algorithms [19]; the idea is either to provide single view depth estimation or directly computing frame-to-frame local feature correspondences.

Our proposed method shares some common traits with SLAM. Both approaches have a module dedicated to inferring the motion of the ego-vehicle: IPR by tracking the 2D ground plane and SLAM by tracking the whole 3D environment. Despite this similarity, the goal of the two methods is very different since we want to retain exactly what SLAM discards, i.e. model the dynamics of other vehicles rather than reconstructing ego-motion and the static environment. Indeed SLAM could serve as a ground motion estimator in our pipeline. Nonetheless SLAM algorithms require internal calibration parameters, while our approach is suitable for any RGB sequence.

3 Iterative Plane Registration

Iterative Plane Registration (IPR) is an procedure to track the ground plane in a video and obtain a series of homographies that can transform points across different frames. We refer to IPR as a meta-algorithm since it outlines a generic algorithmic procedure based on different computer vision modules, without relying on any specific model or architecture. The modules composing the Iterative Plane Registration meta-algorithm are shown in Fig. 2 and are the following: object detector, multiple target tracker, semantic segmentation model, local keypoint detector and descriptor and homography estimator.

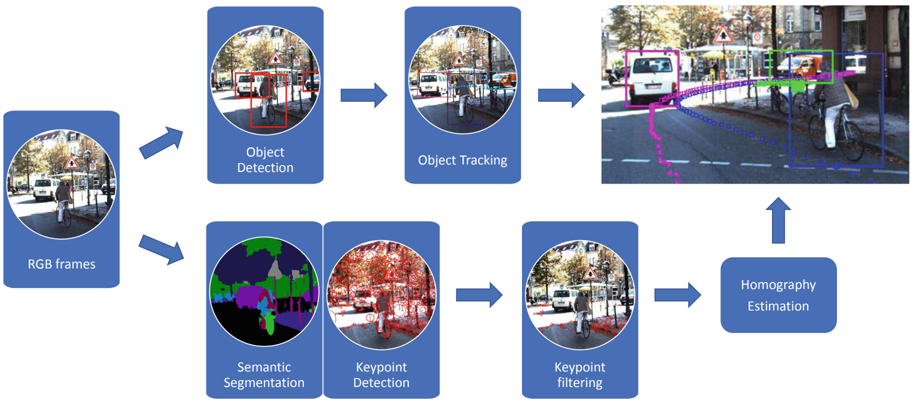

Fig. 2. Iterative Plane Registration pipeline. Objects are detected and tracked. The ground plane is tracked with an homography estimated through keypoints detected in the image and filtered with the semantic segmentation. Chains of homographies are estimated to warp the position of the objects across frames.

The advantage of defining IPR as a meta-algorithm is that, thanks to its highly modular nature, it can be easily updated by replacing its building blocks keeping up with future state of the art advancements.

Object Detection and Tracking. Agents have to be localized in each frame and tracked across the whole video. To this end, we use Mask-RCNN [13] as object detector and the bounding box association algorithm proposed in [8] as multiple target tracker. The method matches bounding boxes in consecutive frames according to their intersection over union and thus generates spatio-temporal tubes enclosing the objects. To ensure an accurate matching, bounding box future positions are predicted using dense optical flow [9] to compensate object and ego motion. To be able to detect relevant objects in an urban scene, we use a Mask-RCNN model pretrained on MS-COCO [16] and we track only objects which are relevant to our task, i.e. objects labeled by the detector as *car, person, bicycle, motorbike, truck* or *train.*

Semantic Segmentation Based Keypoint Detection. In order to estimate reliable transformations to map the ground plane from a frame to another, we extract local keypoints from the scene and filter them using the output of a semantic segmentation method. As keypoints we use SIFT [17], masking the input image with the semantic segmentation provided by DeepLab v3+ [6]. Since we want to obtain keypoints belonging to the ground plane, we retain only the ones centered in pixels labeled as *road* or *sidewalk*, independently of the scale of the detected keypoint.

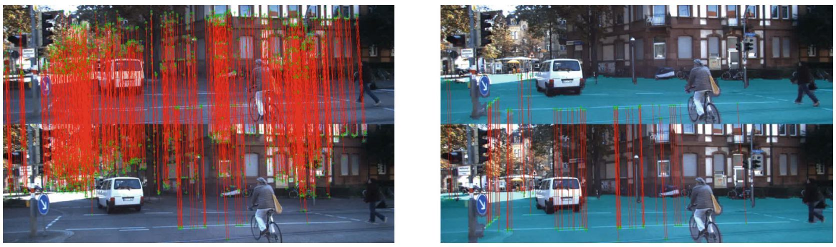

Fig. 3. Keypoint matching between two frames. When all the keypoints are used (left), correspondences are found all over the scene. When keypoints are filtered with the semantic segmentation (right) matches are reliably found only on the ground plane.

Homography Estimation. In the following experiments we use SIFT since they are the best trade-off in terms of stability, repeatability and speed. Any other local feature could be employed in principle. SIFT keypoints and their associated descriptors are used to estimate homographies between frames. This is done using Random Sample Consensus (RANSAC) [10] between the two set of matching keypoints L_{t_i} and $L_{t_{i+1}}$, belonging to frames at time t_i and t_{i+1}. RANSAC finds the transformation $\mathbf{H}_{t_i}$ that maps keypoints $k_{t_i}^j \in L_{t_i}$ in their correspondent ones $k^j_{ti+1} \in L_{t+1}$ in the next frame, rejecting outlier correspondences. The semantic segmentation filter over all the keypoints in the scene is necessary since we only want to model the planar homography for the pixels belonging to the actual road. By doing so we are working unders the assumption that the ground can be locally approximated by a planar surface. Without relying on the semantic segmentation we cannot establish the correct correspondences between keypoints, yielding to an incorrect homography. Figure 3 shows an example of matched keypoints between two frames, with and without the segmentation mask. It can be seen that without segmenting the scene, it is likely to establish correspondences between other planar surfaces, such as buildings, which are often rich in texture and therefore keypoints.

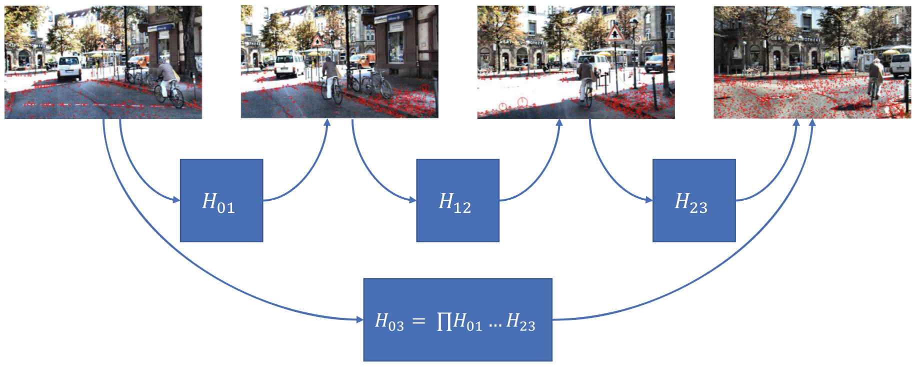

Fig. 4. Chained homographies to warp points lying on the ground plane across frames.

Trajectory Projection. To generate holistic trajectories of other agents in a given frame F_{t_i}, we project their positions in other frames F_{t_j} using a chain of homographies from t_j to t_i:

$$H = \prod H_{t_k} \qquad \forall t_k \in [t_j, t_i]. \tag{1}$$

This procedure is also depicted in Fig. 4.

To map points forward in time we use the homographies estimated between pairs of consecutive frames, while to map points backward in time we use inverse homographies. Since each homography can only transform points belonging to the ground plane we cannot warp bounding boxes. We therefore project only the lower edge middle point of a bounding box, which is guaranteed to lie on the ground plane. An example of generated trajectories is depicted in Fig. 1.

Combining chains of homographies may lead to incorrect results due to numerical instability. To determine whether an homography is valid or not, we check the determinant of the transformation matrix [12]: $\det(\mathbf{H}) > 0$. If an homography is not valid, we interrupt the chain of homographies and we stop projecting the trajectories, marking the remaining portion as invalid.

Algorithm 1. Iterative Plane Registration

```
Input: RGB video sequence $F_{t_i}, t_i \in [t_0, t_{end}]$
Output: Homography set
1: Initial timestep $t_0$.
2: while $t_i < t_{end}$ do
3:     Apply semantic segmentation algorithm (e.g. DeepLab [6]) to frame $F_{t_i}$, obtaining a pixel-wise labeling $S^c_{t_i}, c \in \{\texttt{'road'}, \texttt{'car'}, \texttt{'sidewalk'} \ldots\}$.
4:     Extract local keypoints $L_{t_i}$ (e.g. SIFT [17]) from $F_{t_i}$.
5:     Discard keypoints not laying on the ground plane based on the semantic segmentation: $L'_{t_i} = \{k \in L_{t_i}\ \texttt{s.t.}\ S_{t_i}[k_x, k_y] \in \{\texttt{'road'}, \texttt{'sidewalk'}\}\}$
6:     Estimate homography to map the ground between frames $F_{t_{i-1}}$ and $F_{t_i}$: $\mathbf{H}_{t_{i-1}t_i} = \texttt{RANSAC}(L'_{t_{i-1}}, L'_{t_i})$
7:     $t_i = t_{i+1}$;
8: end while
9: return $\{\mathbf{H}_{t_i}\}$
```

4 Results

The Iterative Plane Registration algorithm can be used on any driving video taken from a dashcam since it requires no annotation. To provide an evaluation of the method, we generated trajectories for all LIDAR annotated videos in the KITTI tracking training set [11]. To evaluate how accurately we register the ground plane, we turn off the detection and tracking modules and consider annotated trajectories instead. Since each trajectory is annotated as a collection of 3D bounding boxes, we warp across frames the center of their lower face. Once the holistic trajectories are obtained, we project them in the LIDAR metric coordinate system using a frame to world homography. Whereas projecting points from LIDAR to frame can be done by changing coordinate system and using the camera projection matrix P, the opposite is not as1 straightforward since P is not invertible. To this end we estimate an homography between the pair of points belonging to the ground plane in the two reference systems. Differently from what happens in the IPR pipeline, we do not need to detect and match keypoints to estimate the homography since there is a direct correspondence between frame pixels and LIDAR points. We only need to filter the points by taking only the ones belonging to the ground plane, which can be done with the semantic segmentation of the scene [6]. The frame to world transformation allows us to project the estimated trajectories in the LIDAR metric reference system and to compare them with the ground truth, obtaining an error in meters (Fig. 5).

Figure 6 (left) shows the distribution of samples, i.e. individual points, as a function of the temporal offset from the current frame and the $\mathcal{L}_2$ distance from the ground truth. Most of the samples have a negligible error since almost half of the points lie in a 5 m radius from the target. Increasing the temporal offset, points estimates become less precise as an effect of error propagation when combining long chains of homographies (Algorithm 1). Furthermore some

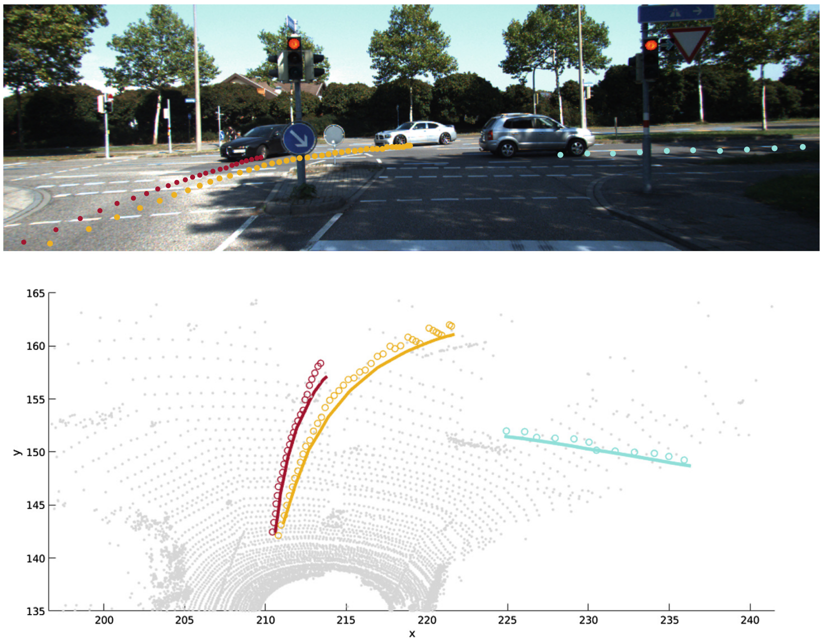

Fig. 5. Generated trajectories in the frame reference system (top) and comparison with ground truth in the LIDAR metric reference system (bottom).

samples exhibit high errors, which are mainly caused by instabilities in warping points far away from the camera, as shown in Fig. 6 (right). On the other hand, it has to be noted that the most relevant signal is in the first seconds ahead. We consider the most useful time span for training a prediction algorithm to be 12 s in the future. Therefore we report in Fig. 7 the distribution of errors for all points in an 12 s horizon.

We also analyze the error in function of distance from the sensor. As can be seen in Fig. 6, below 50 m of distance errors are mostly below 5 m. This distance can be regarded as a common visibility horizon in urban scenarios, with junctions, curved road and occlusions due to traffic. Consider that the KITTI LIDAR sensor reach is 120 m but we can, in certain cases obtain farther distances by ground plane registration.

Another interesting evaluation concerns the number of trajectories we are able to obtain. To this end we ran the Iterative Plane Registration algorithm on the whole KITTI tracking dataset (both train and test). We generate trajectories up to 12 s (120 frames at 10 FPS), both in the past and in the future. According to the determinant criterion explained in Sect. 3, parts of tracks generated by invalid homographies are discarded.

In Fig. 8 we show the number of obtained trajectories, as a function of past and future length. Both valid and invalid trajectories are shown. Interestingly enough, invalid homographies concern mostly past trajectories. This is due to

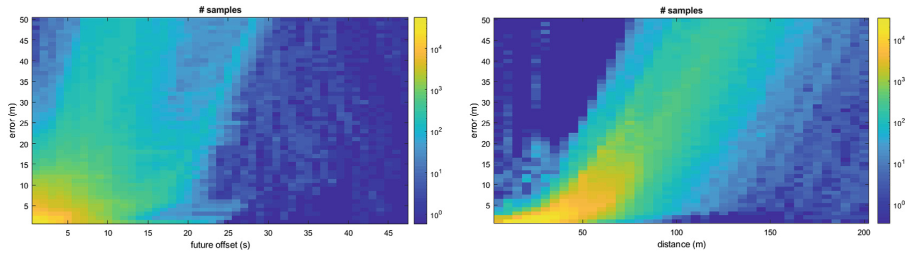

Fig. 6. Distribution of errors over samples (individual points) as a function of future offset (left) and distance (right).

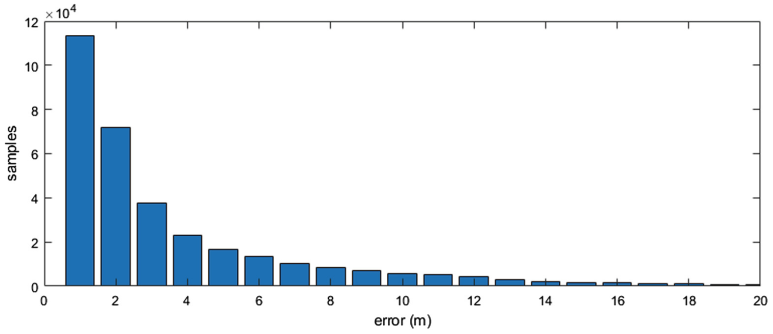

Fig. 7. Error distribution for points up to 12 s in the future.

the fact that from a car perspective the ground plane is observed from the car ahead, therefore the estimated homographies will be less precise in the portion of the plane behind the observer, which is often where the other agents lie in past time-steps.

Despite this, we are able to generate a surprisingly high number of trajectories, both in past and future directions. On the KITTI tracking dataset we obtain approximately 55K and 73K samples for the training and test set respectively, with an average of 6.7 trajectories per image. Note that the whole KITTI tracking dataset only contains 896 training trajectories. The different nature of our trajectories allows us to obtain a much higher number of samples both for training and for testing. This high number of trajectories stems from the fact that we are generating a new holistic trajectory from each frame in which the agent is observed. Whereas these trajectories are correlated since they represent the same agent, the resulting series of points is quite different due to camera motion and context variability. Overall, this acts as a form of data augmentation over existing trajectories, multiplying the occurrences of a trajectory for each frame in which the object is present.

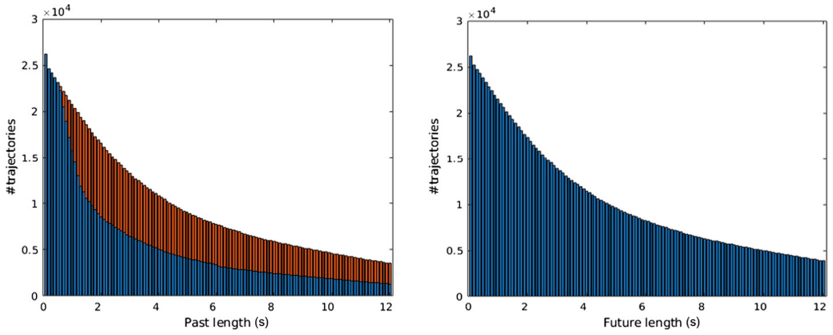

Fig. 8. Number of obtainable trajectories on the KITTI dataset (*train*) as a function of past and future number of frames. Both valid (blue) and invalid (red) trajectories are shown. (Color figure online)

5 Conclusions

In this paper we presented the Iterative Plane Registration meta-algorithm, a procedure for collecting holistic trajectories of agents in urban scenarios without requiring any prior annotation. The generated trajectories are composed by past, present, and future positions, all projected into a single frame context. Thanks to Iterative Plane Registration we are able to obtain an extremely high number of trajectories which can be used to train predictive models for autonomous driving vehicles.

Acknowledgements. This work has been developed within a collaboration with IMRA Europe. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social LSTM: human trajectory prediction in crowded spaces. In: Proceedings of CVPR, pp. 961–971 (2016)
2. Badue, C., et al.: Self-driving cars: a survey. arXiv preprint arXiv:1901.04407 (2019)
3. Berlincioni, L., Becattini, F., Galteri, L., Seidenari, L., Del Bimbo, A.: Road layout understanding by generative adversarial inpainting. arXiv preprint arXiv:1805.11746 (2018)
4. Bresson, G., Alsayed, Z., Yu, L., Glaser, S.: Simultaneous localization and mapping: a survey of current trends in autonomous driving. IEEE Trans. Intell. Veh. **20**, 1 (2017)
5. Caesar, H., et al.: nuScenes: a multimodal dataset for autonomous driving. arXiv preprint arXiv:1903.11027 (2019)
6. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)

7. Cordts, M., et al.: The cityscapes dataset. In: Proceedings of CVPRW (2015)
8. Cuffaro, G., Becattini, F., Baecchi, C., Seidenari, L., Del Bimbo, A.: Segmentation free object discovery in video. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 25–31. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_4
9. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 363–370. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45103-X_50
10. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
11. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: Proceedings of CVPR (2012)
12. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2003)
13. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of ICCV (2017)
14. Huang, X., et al.: The ApolloScape dataset for autonomous driving. In: Proceedings of CVPRW, pp. 954–960 (2018)
15. Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H., Chandraker, M.: Desire: distant future prediction in dynamic scenes with interacting agents. In: Proceedings of CVPR (2017)
16. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
17. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision **60**(2), 91–110 (2004)
18. Ma, Y., Zhu, X., Zhang, S., Yang, R., Wang, W., Manocha, D.: TrafficPredict: trajectory prediction for heterogeneous traffic-agents. arXiv preprint arXiv:1811.02146 (2018)
19. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans. Rob. **33**(5), 1255–1262 (2017)
20. Pellegrini, S., Ess, A., Schindler, K., Van Gool, L.: You'll never walk alone: modeling social behavior for multi-target tracking. In: Proceedings of ICCV (2009)
21. Robicquet, A., Sadeghian, A., Alahi, A., Savarese, S.: Learning social etiquette: human trajectory understanding in crowded scenes. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 549–565. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_33
22. Srikanth, S., Ansari, J.A., Sharma, S., et al.: Infer: intermediate representations for future prediction. arXiv preprint arXiv:1903.10641 (2019)
23. Tang, J., Folkesson, J., Jensfelt, P.: Geometric correspondence network for camera motion estimation. IEEE Rob. Autom. Lett. **3**(2), 1010–1017 (2018)
24. Tateno, K., Tombari, F., Laina, I., Navab, N.: CNN-SLAM: real-time dense monocular SLAM with learned depth prediction. In: Proceedings of CVPR (2017)
25. Yu, F., et al.: BDD100K: a diverse driving video database with scalable annotation tooling. arXiv preprint arXiv:1805.04687 (2018)