Supplementary Material - Unsupervised Learning of Shape-Motion Patterns for Objects in Urban Street Scenes

Dirk Klostermann
dirk.klostermann@rwth-aachen.de
Aljosa Osep
osep@vision.rwth-aachen.de
Jörg Stückler
stueckler@vision.rwth-aachen.de
Bastian Leibe
leibe@vision.rwth-aachen.com

Computer Vision Group, Visual Computing Institute, RWTH Aachen University
Aachen, Germany

Contents

1 Shape Features

Our shape clustering and matching approach requires repeatable descriptors represented by the GCTs. To retrieve such features, we take care that the GCTs are aligned consistently in position and orientation with respect to a reference frame in the object. In the following, we provide details and motivations on our design choices for the shape features.

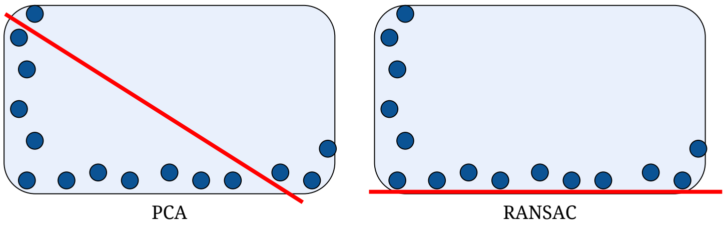

Figure 1: Comparison of the orientation estimates using PCA or RANSAC. An object is shown from a bird's eye perspective. The blue circles indicate the surface points. The red line visualizes the orientation direction which would be estimated by each of the algorithms.

1.1 Reference Position

GCTs are designed to model the shape of objects that only change position and orientation in the ground plane. Thus, we extract a reference vertical axis in the center of the object as the vertical axis of its GCT. We place this reference axis at the center of the 3D bounding box of the object measurements.

The GCT is integrated from object measurements of multiple frames over time, such that the 3D bounding box changes with the measurements and a realignment of the center axis becomes necessary. Since the recentering of the model is computationally expensive, we recenter the GCT only, if the new center diverges more than a certain distance (10 cm in our experiments) from the current position of the vertical axis. By this, we rarely recenter and the recentering does not affect the performance of the tracker significantly.

1.2 Reference Orientation

Since objects are observed with varying orientation, we aim at describing trajectories and GCT shapes in an orientation-invariant way. To this end, we estimate a reference orientation in the ground plane from the object measurements and normalize trajectory and shape with respect to this orientation. Notably, the estimated orientation does not need to be an interpretable orientation like a moving or viewing direction, but needs to be repeatable for the shape category. We identified the following possible ways for orientation estimation:

1. Direction of motion: In related work, objects are often defined as moving parts of the scene [3, 6] and the direction of motion is used as a reference. In our approach, however, we also want to model static objects such as parked cars or standing pedestrians.
2. Principal component analysis (PCA): We can estimate the main axis of the data points contained in the GCT with PCA. The main axis should be similar for objects of the same category. This approach only considers shape and hence it is applicable for static and dynamic objects. However, this estimation is inaccurate for partially observed objects because the main axis will depend on the distribution of observed parts (see Figure 1).
3. Robust plane fit: With the RANSAC algorithm we can fit a dominant plane to the object measurements to retrieve the object orientation. Compared to PCA, this has the advantage that such a plane can still be found for partially observed objects. For example, in Figure 1, we fit a plane to the side of the car. This approach has also been used by Teichman and Thrun [7] and is used in our method.

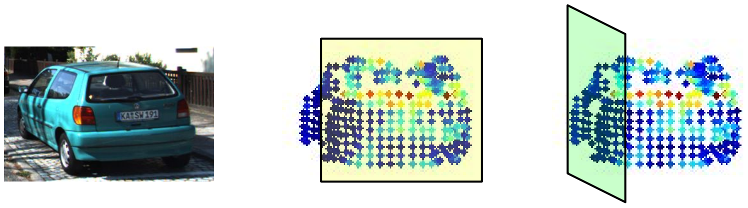
Figure 2: Depending on the view point on the object, different planes can be fit to the object measurements.

While the RANSAC orientation estimate is less prone to drift as the PCA estimate, it still may find one of multiple possible plane fits. For example, for the orientation estimate of the car shown in Figure 2, we can fit a plane which is perpendicular to (yellow plane) or along (green plane) its forward direction. Even worse, for cylindrical objects the orientation estimation could be arbitrary, but such objects rather rarely occur in urban street scenes. Typically for pedestrians which are close to cylindrical objects, we can extract one of two main axes (frontal or sagittal directions) from the measurements. We observed that most objects underlie an ambiguity of four different orientations. We can compensate for this ambiguity by training on enough data such that there are good motion models for each of these cases.

2 Distance Measures

In this section we describe further details on GCT distance measures and the reasoning behind choosing appropriate default values for unobserved parts of the GCT. Mitzel [4] proposes to compare median-based GCTs using the Euclidean distance and histogram-based GCTs using histogram intersection. A single orientation bin can be compared by

$$d_M(x,y) = \|x-y\|_2\,, \qquad (1) \qquad\qquad d_H(x,y) = 1 - \sum_{h=0}^{B} min\{x(h),y(h)\} \qquad (2)$$

2.1 Default Distance (DD)

Since the objects are only observed partially, several bins typically remain unobserved. Without further handling, unobserved bins would be represented with a median distance of zero or an empty histogram, respectively. This biases the distance measure significantly, causing large distances between GCTs of similar shapes with missing overlap of observed volume.

We address the problem of unobserved bins by setting them to a default distance or histogram content as in [4]. We denote the default distance by $\gamma \in \mathbb{R}$ and the default histogram by $\gamma \in \mathbb{R}^H$, hence, we use the GCTs

$$s'(i,j) = \begin{cases} s(i,j), & \text{if } s(i,j) \neq \emptyset \\ \gamma, & \text{otherwise.} \end{cases} \qquad (3)$$

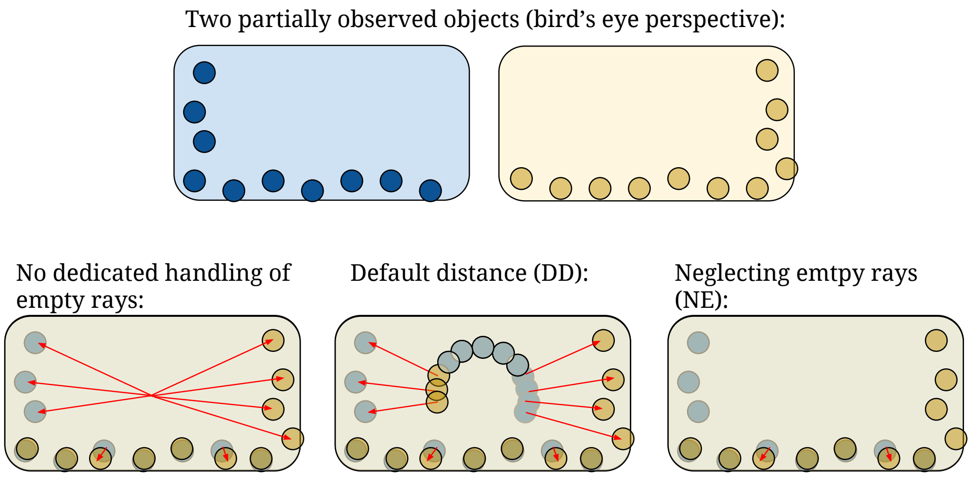
Two partially observed objects (bird's eye perspective):
No dedicated handling of empty rays:
Default distance (DD):
Neglecting emtpy rays (NE):

Figure 3: Visualization of the error for comparing partially observed objects through either no dedicated handling, DD or NE. The red arrows visualize the resulting error.

2.2 Neglecting Empty Bins (NE)

Alternatively to substituting empty bins in a preprocessing step, we can neglect the empty bins in the distance calculation. This way, only the bins which are filled in both GCTs influence the distance.

We can adapt the distance function $d(s,s')$ between GCTs s, s' to reflect this concept:

$$d_{NE}(s,s') = \alpha \sum_{i=0}^{N} \sum_{j=0}^{M} \begin{cases} d(s(i,j), s'(i,j)), & \text{if } s(i,j) \neq \emptyset \wedge s'(i,j) \neq \emptyset \\ 0, & \text{else} \end{cases} \tag{4}$$

where $s(i,j) \neq \emptyset$ iff the bin $s(i,j)$ is not empty. α denotes the normalization factor:

$$\alpha = \frac{1}{(NM) - E(s,s')}. \tag{5}$$

$E(s,s')$ denotes the number of bins which are filled in both GCTs:

$$E(s,s') = \sum_{i=0}^{N} \sum_{j=0}^{M} \begin{cases} 1, & \text{if } s(i,j) \neq \emptyset \wedge s'(i,j) \neq \emptyset \\ 0, & \text{else.} \end{cases} \tag{6}$$

2.3 Comparison of Distance Measures

With the DD approach, we can distinguish shapes of different size better than by neglecting empty bins. This is especially useful in urban street scenes where many objects types vary in heights as for example trucks, cars or dogs. However, the DD approach also penalizes empty bins if they are due to unobserved parts of an object. Neglecting empty bins, on the other hand, ignores the parts which were not jointly observed in the two shapes. With the default distance value in the DD approach, we can control the penalization of empty bins. In the following, we give further insights and an empirical analysis for both distance measures.

Figure 3 visualizes the resulting error for comparing partially observed objects if either empty bins are not handled in a dedicated way, if they are handled by the DD approach or if they are handled by the NE approach. Without dedicated handling, empty bins fully add error terms to the distance metric and we cannot control their contribution. With the DD approach we can change their influence on the distance metric by modifying the default distance or histogram values. Lastly, with the NE approach we focus on the filled bins only.

In Figure 4 the performance of different configurations are visualized. In this setting the clustering parameters are optimized for achieving the best value on the $V_{0.1}$ metric. The $V_{0.1}$ metric is a modification of the V measure, which is a combination of homogeneity and completeness. Its parameter β can be set to favour homogeneity [5]:

$$V_\beta = \frac{(1+\beta) \cdot homogeneity \cdot completeness}{(\beta \cdot homogeneity) + completeness}. \tag{7}$$

The combination of median features (ME) and using a default distance (DD) scores best on the $V_{0.1}$ metric but produces an unreasonable number of clusters. The combinations HI-DD and ME-NE achieve a $V_{0.1}$ metric similar to ME-DD with a reasonable number of clusters. Hence we favour these combinations in our evaluation. While the variation in these configurations seems to be rather small, the difference in $V_{0.1}$ between ME-Euclidean and ME-DD amounts to almost 0.1 which is large in matter of the $V_{0.1}$ metric. Note that HI-DD and ME-NE achieve a larger value on the $V_{0.1}$ metric than ME-Euclidean and a similar score to Histogram Intersection but produce less clusters at the same time.

3 Trajectory Features

Trajectory information can be used as features in various ways. We compare four different representations of trajectory information (Figure 5):

1. Current motion (CM): At each time step, we could estimate the current acceleration, velocity and angular velocity of the object and use it as a feature. With this information, a quadratic extrapolation of the object motion could be made as in [1].
2. Trajectory characteristics (TC): We can define a local window around the current time step or analyse the whole trajectory up to the current time step. By this, we could describe the analysed trajectory part by the minima, maxima and mean of velocity, acceleration and angular velocity for as for example in [8]:

$$(v_{min}, v_{max}, v_{mean}, a_{max}, a_{mean}, rot_{max}, rot_{mean}, rot_{sum}).$$

   Here, v denotes the velocity, a the acceleration and rot the angular velocity, while rot_{sum} denotes the sum of all orientation changes in the local window of the trajectory.
3. Relative position (RP): Kooji et al. [2] model the trajectory relative to the camera frame (see Figure 5). Note, that ego-motion is compensated as well. This way, the motion model strongly depends on the position of the object relative to the camera. This can be advantageous for deciding if, for example, a pedestrian will cross the street, because it implicitly models the distance of the object to the curb and the distance to the observing car.
4. Relative trajectory (RT): Alternatively, the reference frame can be placed individually for each tracked object at the current position of the object. We rotate the coordinate system according to the estimated orientation in order to align the trajectories of the observed objects. This representation is not biased by the position and hence it is

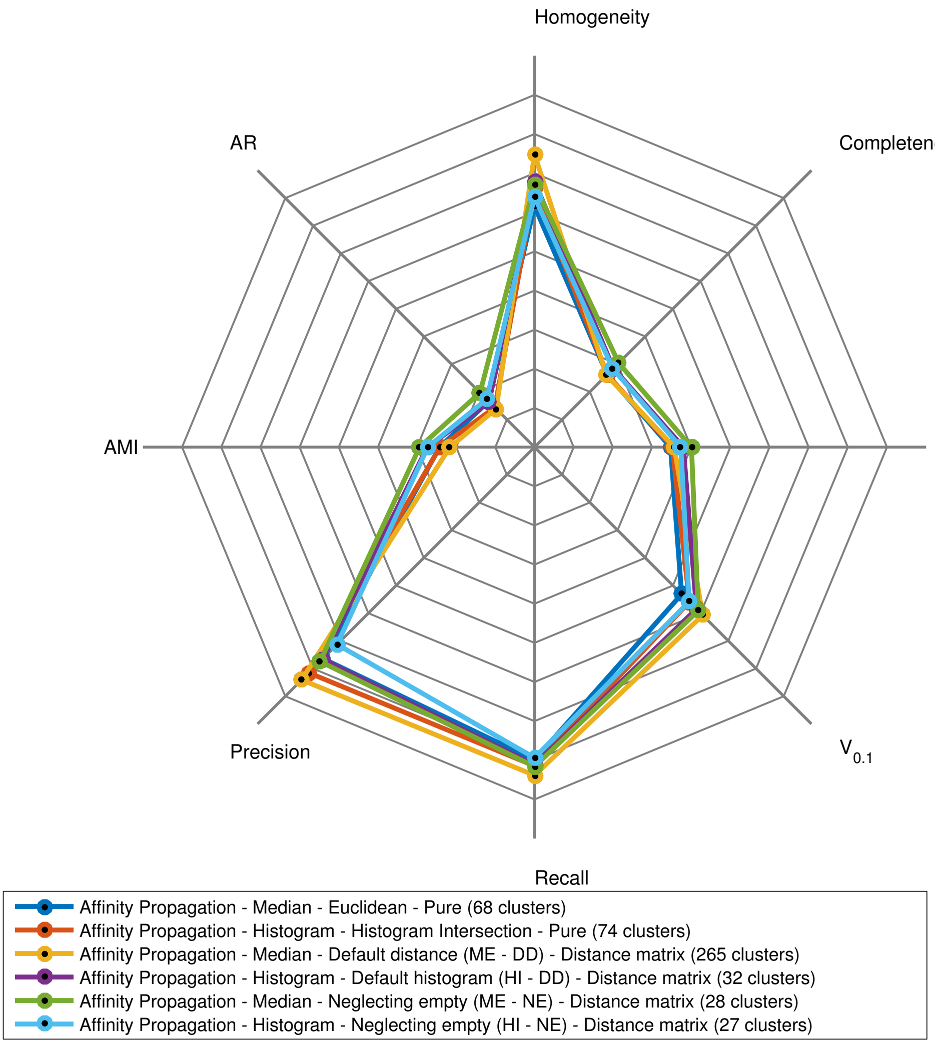

Figure 4: Comparison of different configuration. In every case Affinity Propagation is used as clustering algorithm and the features are transformed to a distance matrix as preprocessing step.

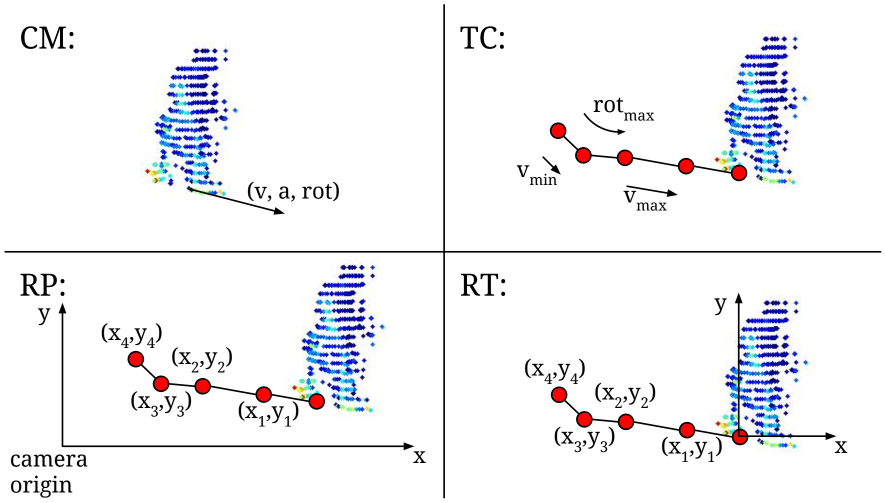

Figure 5: Visualization of trajectory representations. Red dots visualize the position of the object.

	K-Mean	Mean Shift	Affinity Propagation	DBSCAN
Cluster shapes				
- Spherical	✓	✓	✓	✓
- Convex		✓	✓	✓
- Non-convex				✓
Cluster representative	(✓)	(✓)	✓	
Number of clusters specified	✓			

Table 1: Comparison of the introduced clustering algorithms

more general. It also enables us to model constraints for the velocity, acceleration and angular velocity of the tracked object.

We choose RT, since it models the past and future trajectory directly and flexibly. Furthermore, it is independent of the relative position of the tracked object.

4 Clustering Algorithm Comparison

We analyse the performance of four clustering algorithms on the task of shape-motion pattern clustering: K-Means, Mean Shift, Affinity Propagation (AP) and DBSCAN. In Table Table 1 we display key characteristics of each algorithm.

We evaluate and compare the performance of the clustering algorithms based on clustering metrics. Figure 6 visualizes the performance of the four clustering algorithms. We observe that the performance on the different metrics strongly relates with the number of found clusters. Mean Shift tends to group large objects (cars, vans and trucks) and small objects (pedestrians and cyclists). Accordingly, it performs best on metrics which favour completeness. Affinity Propagation and DBSCAN distribute the instances of each class over multiple clusters. This results in a higher homogeneity but decreases completeness. The metric $V_{0.1}$ combines homogeneity and completeness by weighting homogeneity 10 times

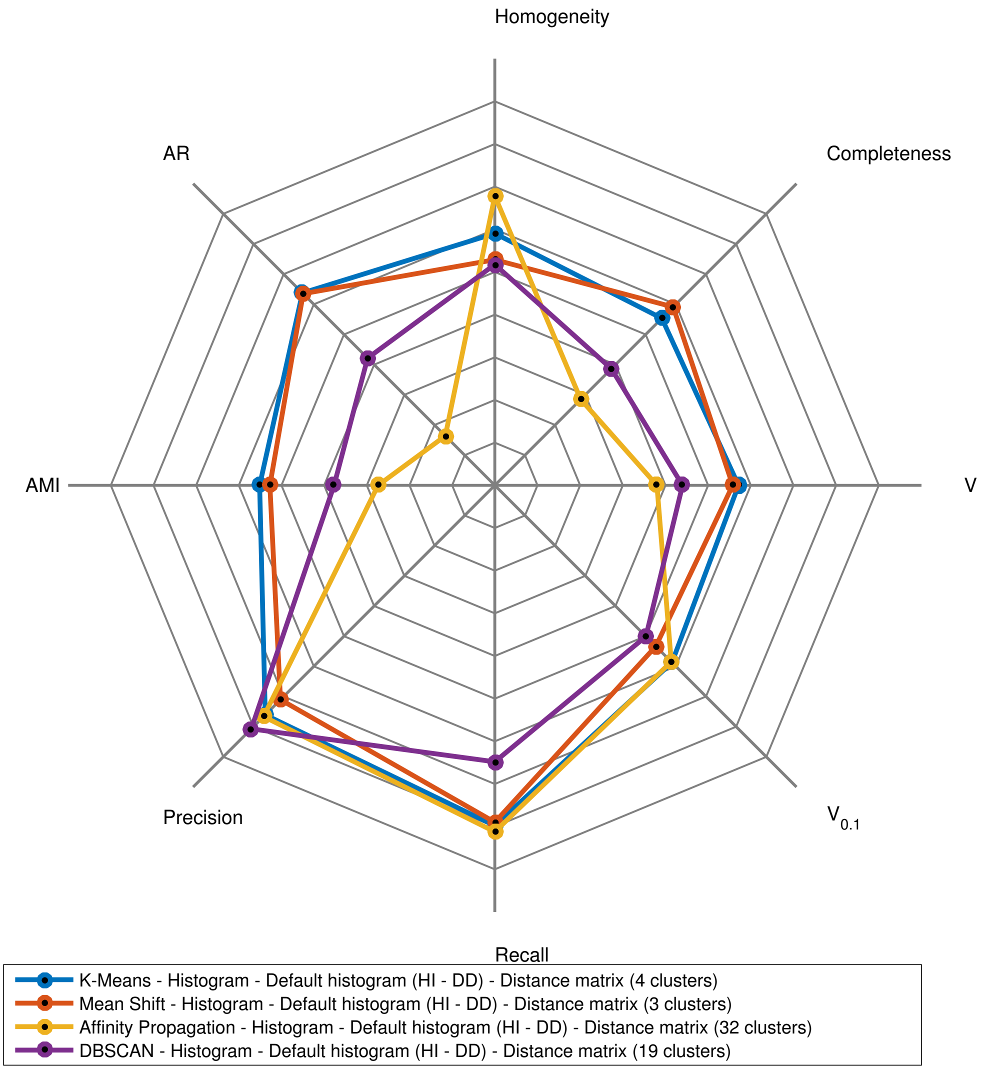

Figure 6: Comparison of the four different clustering algorithms. In every case the GCT was represented as histogram and empty bins were filled by a default histogram.

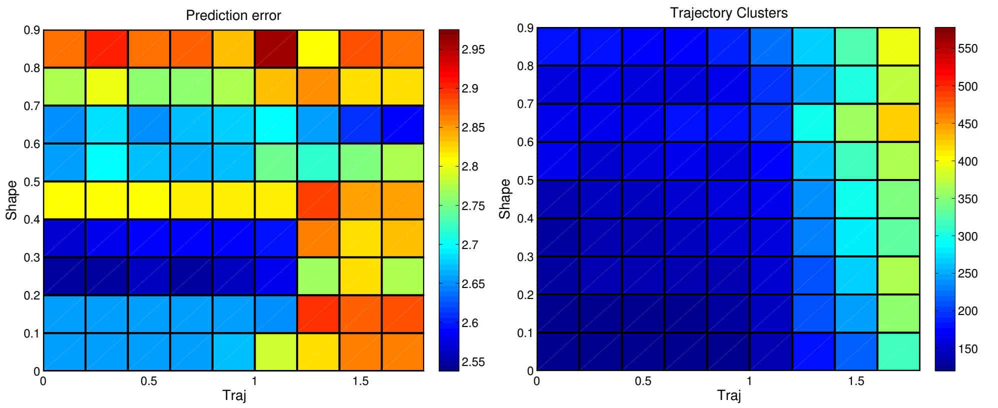

Figure 7: Parameter estimation for the SMP clustering.

stronger than completeness. Since Affinity Propagation and K-Means achieve the largest value of $V_{0.1}$, we can assume that the clusters of Affinity Propagation and K-Means fit best to the training data. Depending on the application we need to decide which type of clustering is desirable. K-Means creates a better clustering in terms of completeness, while Affinity Propagation creates the best clustering in terms of homogeneity among the four clustering algorithms. We argue that in most applications, homogeneity is more desirable than completeness: If the instances of one class are distributed over multiple clusters, the various clusters could still be easily annotated by an expert. Hence, we use Affinity Propagation in our approach.

5 Parameter Estimation

We cluster the shapes and the trajectory features in each shape cluster with Affinity Propagation (AP). The clusters of AP mainly depends on the preference parameter. This parameter models for each instance how likely it is that this instance will be a representative and hence forms a cluster by its own. Thus, by increasing the preference parameter, the number of clusters increases. With a grid search approach, we try a range of preference parameters for the shape and the trajectory clustering step. We vary the default preference value (median of the distances between the instances) by a factor λ. Figure 7, left, shows the resulting prediction error. The right figure shows the number of generated SMPs in each case. In our experiments, we choose a preference parameter of $\lambda_{shape} = 0.3$ for shape and $\lambda_{traj} = 0.8$ for trajectories. Analogously, we estimate the optimal parameter for the pure motion approach.

6 Dataset Augmentation

The amount of training data is crucial for the performance of the algorithm. For each object in the test set, many similar objects are needed in the training set to cover the ambiguities of the orientation estimation and the variability resulting from partial observations well.

The KITTI training dataset contains only about 500 meaningful training examples of shapes and trajectories. From each trajectory we extract up to 10 feature vectors. Extracting more feature vectors from each track reduces the variability within the feature space and

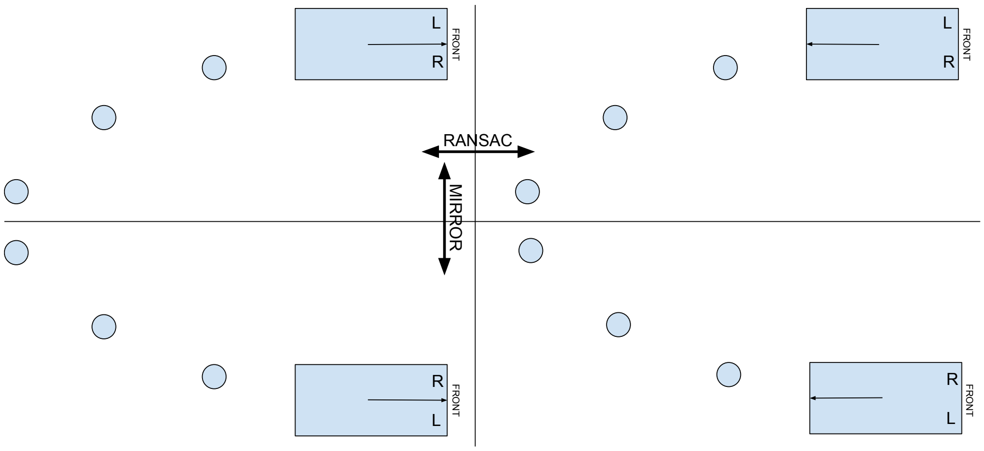

Figure 8: The number of features can be increased by creating artificial shape/trajectory combinations based on the existing ones. *(left, top)* displays an object together with the three last positions. *(right, top)* Due to the ambiguity introduced by using RANSAC as orientation estimation the orientation can also be turned by 180 degrees. *(left, bottom)* With the assumption that shape and trajectory are axis symmetric in the orientation axis we can create another artificial copy. This represents, for example, that a car which does a left turn can also do a right turn. *(right, bottom)* Lastly, both effects can be combined.

does not result in new objects from new viewpoints. The limited number of only 500 feature vectors is conditioned to the trajectory representation. We represent a trajectory by the offset to the 20 past and 20 future positions of the tracked objects. Thus, an object needs to be tracked for 4 seconds to be usable as training instance. Id switches or shortly interrupted tracks limit the number of suitable objects further.

We augment the limited training data to increase the number of training instances. First, we can exploit the RANSAC ambiguity to generate for each existing training instance a second one. For the orientation estimation with RANSAC, we fit a plane to the point cloud and use the normal vector of this plane as orientation estimation. Hence, a GCT for which the orientation is changed by 180 degree is also a valid training instance. In Figure 8 the both upper examples visualize this augmentation. We copy the original GCT and the trajectory (left), and rotate the orientation by 180 degree (right).

Next, we can assume that objects and their trajectories can be mirrored due to the inherent symmetry of the typical objects in street scenes. For example, if a car drives a left turn it can also drive a right one. In the lower right part of Figure 8 we mirror the original object together with its trajectory along the orientation axis. Last but not least, we can achieve a fourth example from the training instance by combining the augmentation due to RANSAC and mirroring.

The augmented dataset is four times larger (about 2000 instances instead of 500 instances). Despite both augmentations are artificial they result in valid shapes and trajectories which we could detect in a street scene. Furthermore, we increase the variability of the training features with the augmented instances since they add instances from a different view point.

References

[1] Joshua Joseph, Finale Doshi-Velez, Albert S Huang, and Nicholas Roy. A bayesian nonparametric approach to modeling motion patterns. *Autonomous Robots*, 31(4):383–400, 2011.

[2] Julian Kooij. *Generative Models for Pedestrian Track Analysis*. Ridderprint BV, 2015.

[3] Matthias Luber, Kai O Arras, Christian Plagemann, and Wolfram Burgard. Classifying dynamic objects. *Autonomous Robots*, 26(2-3):141–151, 2009.

[4] Dennis Mitzel, Jasper Diesel, Aljoša Ošep, Umer Rafi, and Bastian Leibe. A fixed-dimensional 3d shape representation for matching partially observed objects in street scenes. In *ICRA*, 2015.

[5] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, 2007.

[6] Ye Tao, Rudolph Triebel, and Daniel Cremers. Semi-supervised online learning for efficient classification of objects in 3d data streams. In *IROS*, 2015.

[7] Alex Teichman and Sebastian Thrun. Tracking-based semi-supervised learning. *IJRR*, pages 804–818, 2012.

[8] Alex Teichman, Jesse Levinson, and Sebastian Thrun. Towards 3d object recognition via classification of arbitrary object tracks. In *ICRA*, 2011.