Motion Clustering and Estimation with Conditional Random Fields

Gian Diego Tipaldi and Fabio Ramos

***Abstract*— Moving objects are present in many robotic applications. An accurate detection and motion estimation of these objects can be crucial for the success and safety of the robot and people surrounding it. This paper presents a new probabilistic framework for clustering dependent or relational data, applied to the problem of motion clustering and estimation. While conventional techniques such as scan differencing perform well in many cases, they usually assume that a good pose estimate is available and fail when points belonging to dynamic objects show some overlap in consecutive readings. The technique proposed, CRF-Clustering, by explicitly reasoning about the underlying motion of the object, is able to deal with poor initial motion estimate and overlapping points. Moreover, it is able to consider the dependencies between neighbor points in the scans to reduce the noise in the clustering assignment. The model parameters can be estimated from labeled data in a statistically sound learning procedure. Experiments show that CRF-Clustering is able to detect moving objects, cluster them and estimate their motion.**

I. Introduction

When operating in urban environments, detection and correct motion estimation of cars, people and other dynamic objects can be essential for the safety of the robot and humans nearby. However, only the detection of moving objects is not enough for reliable navigation in complex environments. It is also necessary to estimate the motion pattern of these objects so as to predict their positions ahead in time, and avoid collisions during the decision making process. Combined, these two tasks can be difficult specially considering the variability of motion patterns. For example, people move in a very nonlinear manner and can rotate and translate without restrictions. Cars have a smoother motion pattern but are much faster, therefore being difficult to track when at high speed.

Most of the existing robotic systems used nowadays possess ranging sensors such as laser range finders which can be employed to estimate the robot's motion assuming the environment is static. Techniques such as the Iterative Closest Point (ICP) [20] estimate the motion of the robot by minimizing the residual distance between points in a reference laser scan and associated points in another scan of the sequence. In the presence of dynamic objects, ICP might fail since the presence of spurious dynamic objects can influence the computation of the robot movement if the environment is assumed static. To tackle this problem, current techniques [7], [14], [19] try to detect the parts of the laser scan that are coming from dynamic objects and eliminate them from the robot's motion estimation. Despite

G. D. Tipaldi is with the Social Robotics Lab, Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany

F. Ramos is with the ARC Center of Excellence for Autonomous Systems, ACFR, University of Sydney 2006, NSW, Australia

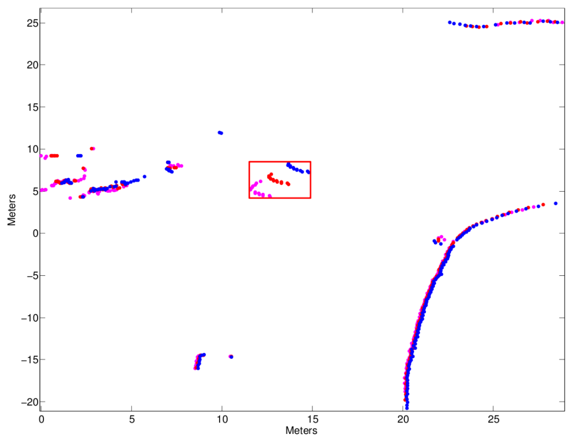


Fig. 1. A sequence of three consecutive scans obtained by our car facing an intersection while another one is passing by. The different colors indicate different scans. The moving car is depicted in the red square.

the importance of the problem, most of the techniques proposed thus far rely deeply into heuristics and manual parameter tuning. This can compromise the robustness of the method when unexpected events take place or when the robot is required to operate in a new environment.

In this paper we present a technique for detecting and estimating the motion of dynamic objects in urban environments based on laser range data. A typical situation is depicted in Figure 1, where the robot is facing an intersection while a car is passing by. We can see that, despite the moving object, the static part of the map is correctly aligned and the car detected (red box). In contrast to scan differencing, we cast the detection problem as a clustering procedure, where associated points in consecutive laser scans are clustered according to their motion patterns. Common clustering techniques such as EM [5] and K-Means [6] assume that points in the data set are independent. Since there is a strong spatial correlation between laser points located nearby, this assumption can severely jeopardize the consistence of these procedures. Frequently, laser returns are generated by rigid objects, thus obeying the Gestalt principles of proximity and common fate [8]. Semi-supervised clustering [4] addresses the independent data assumption by enforcing several constraints among cluster assignments. However, these constraints are usually given by an expert and not extracted from the data. To overcome this limitation, we propose a semi-supervised clustering procedure based on Conditional Random Fields (CRFs) [9]. CRFs are probabilistic graphical models originally proposed for classification of sequential data. These models can be learned discriminately, eliminating the need to multiply conditional distributions with prior distributions as in generative models such as Markov Random Fields (MRFs) and Hidden Markov Models (HMMs). This significantly simplifies the modeling process and allows the specification of more complex constraints to capture particular aspects of the data. Furthermore, our method is able to determine the number of clusters and the corresponding motion patterns while simultaneously computing their parameters.

The paper is organized as follows: previous works on detecting moving objects and estimating their motion patterns are discussed in Section II. We begin introducing our method by first reviewing CRFs in Section III. Our algorithm is detailed in Section IV with discussions on finding the right number of clusters. We provide extensive experimental evaluation on urban data in Section V, where the algorithm is also compared to an existing technique. Conclusions and ideas for future work are presented in Section VI.

II. Related Work

The detection and tracking of moving object (DATMO) problem has been extensively studied [2] for several decades. The problem has been addressed from different scenarios and using different sensors. In terms of related works, we will focus our attention to the detection of moving objects from a moving platform and using a laser range finder.

A first class of algorithms addresses the detection problem only in terms of separating the data into two main clusters: static and dynamic. The dynamic points are then filtered out to obtain a better motion estimation for the moving platform. Hähnel *et al.* [7] presented an EM based approach for detecting moving points in range data. The algorithm maximizes the likelihood of the data using a hidden variable expressing the nature of the points (static or dynamic). This is an offline algorithm which is not suitable for real time applications. Rodriguez-Losada and Minguez [14] showed how data association can be improved for the ICP algorithm in the presence of dynamic objects. They introduced a new metric which better reflects the real motion of the robot. However, their approach does not distinguish moving object from outliers. The approach of Wolf and Sukhatme [19] maintains two separate maps for the static and dynamic part of the environment. The maps are updated using a modified version of the occupancy grid framework which also infers the nature of the points (static or dynamic).

Another class of algorithm focuses also on the object segmentation and tracking. Anguelov *et al.* [1], [3] use simple differencing for detecting the moving points and then apply a modified EM algorithm for clustering the different objects. However, the algorithm needs the number of objects as input and does not consider interactions between neighbor points. In [16], a feature based approach is used to detect the moving objects. These objects are then tracked using a joint probabilistic data association filter (JPDAF). The features used are the local minima of the laser data. While this feature works well in the presence of people, it is not the case of larger moving objects such as cars, buses and so on. Wang *et al.* [18] defined an integrated solution for the mapping and tracking problem: static points are used for mapping while dynamic ones for tracking. The detection and segmentation of dynamic points is based on data differencing and consistency-based motion detection [18]. Points are classified in static and dynamic and clustered in segments. When a segment contains enough dynamic points is considered dynamic. Montesano *et al.* [11] improved the classification procedure described in [18] by jointly solving it in a Bayesian framework. Moreover, they integrated the mapping and tracking within a path planner for indoor navigation. Although, most of these approaches focuses on how to track the different objects under different hypothesis, the detection part is mainly based on different heuristics. The main technique used is based on scan differencing, where points are considered dynamic if there is some inconsistency between two consecutive scans (or a map and a scan). The detection routine is only able to observe the actual position of the object (given a stable reference point) and the velocities are computed by the tracking algorithm.

In this paper we address the problem in a more formal way: points are clustered according to their inherent motion while simultaneously computing their motion parameters.

III. Conditional Random Fields

Conditional Random Fields (CRFs) are undirected graphical models developed for labeling sequence data [9]. CRFs directly model the *conditional* distribution over the hidden variables $\mathbf{x}$ given observations $\mathbf{z}$. Due to this structure, CRFs can handle arbitrary dependencies between the observations $\mathbf{z}$, which gives them substantial flexibility in using high-dimensional feature vectors.

The nodes in a CRF represent hidden states, denoted $\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \rangle$, and data, denoted $\mathbf{z}$. The nodes $\mathbf{x}_i$, along with the connectivity structure represented by the undirected edges between them, define the conditional distribution $p(\mathbf{x}|\mathbf{z})$ over the hidden states $\mathbf{x}$. Let $\mathcal{C}$ be the set of cliques in the graph of a CRF. Then, a CRF factorizes the conditional distribution into a product of *clique potentials* $\phi_c(\mathbf{z}, \mathbf{x}_c)$, where every $c \in \mathcal{C}$ is a clique in the graph and $\mathbf{z}$ and $\mathbf{x}_c$ are the observed data and the hidden nodes in the clique c, respectively. Potentials $\phi_c(\mathbf{z}, \mathbf{x}_c)$ are described by log-linear combinations of *feature functions* $\mathbf{f}_c$, i.e., $\phi_c(\mathbf{z}, \mathbf{x}_c) = \exp\left(\mathbf{w}_c^T \cdot \mathbf{f}_c(\mathbf{z}, \mathbf{x}_c)\right)$, where $\mathbf{w}_c^T$ is a weight vector, and $\mathbf{f}_c(\mathbf{z}, \mathbf{x}_c)$ is a function that extracts a vector of features from the variable values. Using feature functions, the conditional distribution becomes

$$p(\mathbf{x} \mid \mathbf{z}) \quad = \quad \frac{1}{Z(\mathbf{z})} \exp\left\{ \sum_{c \in \mathcal{C}} \mathbf{w}_c^T \cdot \mathbf{f}_c(\mathbf{z}, \mathbf{x}_c) \right\}, \qquad (1)$$

where $Z(\mathbf{z}) = \sum_{\mathbf{x}} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{z}, \mathbf{x}_c)$ is the normalizing partition function.

Inference in CRFs can estimate either the marginal distribution of each hidden variable $\mathbf{x}_i$ or the most likely configuration of all hidden variables $\mathbf{x}$ (i.e., MAP estimation), as defined in (1). Both tasks can be solved using *belief propagation* (BP) [12], which works by sending local messages through the graph structure of the model. Each node sends messages to its neighbors based on messages it receives and the clique potentials, which are defined via the observations and the neighborhood relation in the CRF.

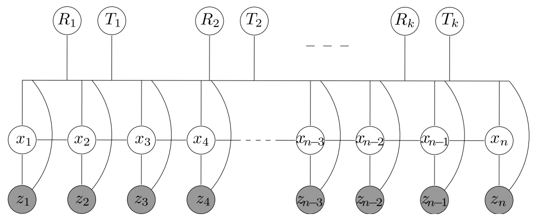


Fig. 2. Graphical representation of the CRF-Clustering model. The hidden states $\mathbf{x}_i$ indicate the cluster number. The observations $\mathbf{z}_i$ correspond to the local features. R_j and T_j indicates rotation and translation of the cluster j.

The weights of the feature functions can be learned discriminatively by maximizing the conditional likelihood of labeled training data, using a numerical gradient algorithm. Unfortunately, this optimization runs an inference procedure at each iteration, which can be intractably inefficient in our case. We therefore resort to maximizing the *pseudo-likelihood* of the training data, which is given by the sum of local likelihoods $p(\mathbf{x}_i \mid \text{MB}(\mathbf{x}_i))$, where $\text{MB}(\mathbf{x}_i)$ is the Markov blanket of variable $\mathbf{x}_i$: the set of the immediate neighbors of $\mathbf{x}_i$ in the CRF graph. In our approach, we use unconstrained L-BFGS [10], an efficient gradient descent method. The key advantage of maximizing pseudo-likelihood rather than the likelihood is that the gradient can be computed extremely efficiently, without running an inference algorithm.

IV. CRF-Clustering

In this section the CRF is extended to address the problem of semi-supervised clustering. Although we describe the methodology for the specific case of motion clustering, the technique is general and can be applied to any clustering problem where the data has some kind of dependence. See [17] for a more general description of CRF-Clustering.

In the following, we will explicitly use the rotational R and translational T parameter of the motion vector. Moreover, in the clustering framework, no distinction is made among the robot and the other moving objects. After clustering, we compute the Mahalanobis distance between the odometry reading and the estimated object motions and classify the motion with the lowest distance as the robot motion.

A. Model Definition

CRF clustering can be understood as a CRF model where local potentials represents distance functions between points and clusters and pairwise potentials represent constraints among different clusters. Figure 2 shows the graph representation of the model (we use blank nodes to represent hidden variables and grey nodes to represent observed variables.). The hidden variables x_i represent the cluster assignment and the hidden variables R_j and T_j represent rotation and translation respectively, for each cluster in the model. The observations z_i are computed from the laser scan pair and are described in the section below.

The inputs to the algorithm are two scans, a reference scan g and second scan s. The objective is to transform parts of s according to the clustering assignments in such a way that the distance between the points in g and the transformed points in s is reduced. Therefore, for every point s_i, a hidden variable x_i is created indicating the clustering assignment. CRF clustering can be trained as a normal CRF using pseudolikelihood. In our experiments, the model parameters were learnt using this technique from a training data set.

B. Local Feature

The current implementation of CRF clustering has one local feature. It is also possible to add more local features describing, for example, the geometry of sets of points or even appearance from vision data. However, to reduce the inference time we consider only the local feature described below:

Distance between laser points: This feature measures the distance between points in the laser scan g and the associated points in the laser scan s after being rotated and translated according to the cluster assignments. As different parts of s are assigned to different clusters this transformation does not preserve the original shape of s. Instead, it tries to break s into parts with the same motion pattern. The equation below describes how this feature is computed for every pair of points i:

$$\mathbf{f}_{\text{dist}}\left(x_i, s_i, g_i\right) = \left\|T_{x_i} + R_{x_i} s_i - g_i\right\|^2 , \tag{2}$$

where x_i defines the cluster j the particular point i belongs to. R_j and T_j are computed from the points assigned to the cluster j by minimizing the sum of distances between points s and corresponding g:

$$R_l, T_l = \arg\min_{R,T} \sum_{s_i, g_i \mid x_i = l} \left\|T + R s_i - g_i\right\|^2 . \tag{3}$$

Fortunately, there is a closed-form solution for the expression above, which can be used to compute R_l and T_l efficiently [20].

C. Pairwise Features

The pairwise features ensure consistency of the clustering assignments by incorporating neighborhood information. For example, if a pair of points i is assigned to cluster l it is very likely that the neighbor pair j will also be assigned to the same cluster. This assumption is true whenever pairs i and j belong to the same rigid object. We define three pairwise features with different properties as follows:

Neighbor Feature: This is a simple pairwise feature returning two possible values. These values are parameters of the model, estimated during learning. More precisely, the feature is defined as:

$$\mathbf{f}_{\text{ng}}\left(x_i, x_j\right) = \begin{cases} \lambda_1, & \text{if } x_i = x_j \\ \lambda_2, & \text{if } x_i \neq x_j \end{cases} \tag{4}$$

where λ_1 and λ_2 are the parameters of the feature.

Weighted Neighbor Feature: This feature is similar to the *Neighbor Feature* except that the output is weighted by the Euclidian distance between the neighbor points. The idea is to capture the notion that neighbor points further away are less dependent than neighbor points nearby.

$$\mathbf{f}_{\text{Wng}}\left(x_i, x_j, s_i, s_j\right) = \begin{cases} \lambda_1/\Delta, & \text{if } x_i = x_j \\ \lambda_2/\Delta, & \text{if } x_i \neq x_j \end{cases} \tag{5}$$

where Δ is the L_2 distance between points s_i and s_j.

Stiffness Feature: This feature tries to enforce stiffness for points that belong to the same cluster. The feature computes the difference of distances between neighbor points before and after the transformation given by the cluster assignment. The idea is that if the points belong to the same cluster, their distances must be preserved after the transformation. This feature can be written as:

$$\mathbf{f}_{\mathrm{st}} = \left\|(s_i - s_j) - \left[(T_{x_i} + R_{x_i} s_i) - (T_{x_j} + R_{x_j} s_j)\right]\right\|^2 \quad (6)$$

D. Inference Procedure

Performing inference in this model is different from performing inference in a normal CRF. Since the values of the observations change with the hidden states normal BP cannot be applied. Instead, we formulate a different message passing procedure where with an initial random cluster assignment, the local features are computed. Messages are then propagated back and forward in the chain model to estimate new values for the hidden variables $\mathbf{x}$. The new cluster assignments $\mathbf{x}$ are used to compute new cluster parameters R and T. Features are recomputed with these new parameters and we iterate this procedure until convergence.

E. Computing the Number of Clusters

One of the interesting properties of our CRF-Clustering model is that it is able to deal with an unknown number of clusters. This is usually an hard problem and is addressed using some *information criterion* (e.g. BIC, AIC, MDL) to trade-off model complexity and fitness to the data. In practice, those methods penalize the likelihood function with an additive term that represents the complexity of the model. In CRF-clustering, the penalizing term is naturally defined by the pairwise potentials, as can be seen if we look at the logarithm of the CRF-Clustering distribution

$$\log p(\mathbf{x}|\mathbf{z}) = \overbrace{-\log Z(\mathbf{z})}^{const.} + \overbrace{\sum_i \mathbf{w}_{\mathrm{dist}}^T \mathbf{f}_{\mathrm{dist}}(\cdot)}^{Likelihood} +$$
$$+ \underbrace{\sum_i \sum_j \left[\mathbf{w}_{\mathrm{ng}}^T \mathbf{f}_{\mathrm{ng}}(\cdot) + \mathbf{w}_{\mathrm{Wng}}^T \mathbf{f}_{\mathrm{Wng}}(\cdot) + \mathbf{w}_{\mathrm{st}}^T \mathbf{f}_{\mathrm{st}}(\cdot)\right]}_{Penalizer} \quad (7)$$

where the weights, $\mathbf{w}$, are learned from training data.

It is worth to notice, however, that the penalizing terms only involve the costraints defined from the graph structure used. In a chain structure, it can happen that points that should belong to the same cluster are not connected, resulting in two different assignments. In order to prevent this, we developed a criterion that allows us to merge similar clusters together. The criterion is inspired by the "effective number of particles" heuristic used in particle filtering, and is called *effective number of clusters*. By using the probability of the cluster assignment of the CRF, p_c, we can compute the effective number of clusters a point belongs to as

$$N_{eff} = \frac{1}{\sum p_c^2}. \quad (8)$$

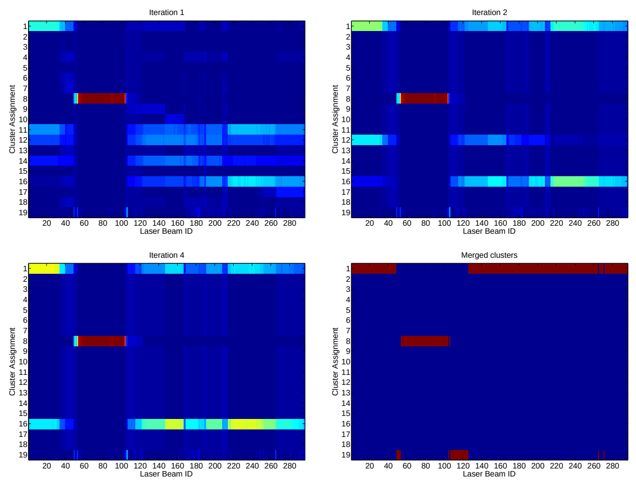


Fig. 3. Convergence of CRF clustering. The first three pictures show the likelihoods of cluster assignments. Dark red represents higher likelihoods and dark blue lower likelihoods. The last picture shows the merging procedure, which correctly merged cluster 1 and 16 together.

We then compute for each cluster the average number of cluster assignments for its points. If it is above 1.7 that cluster is eliminated and its points merged with the second best cluster; the cluster that contains most of the point assignments. Experiments show that we are able to obtain the exact number of clusters, regardless how we initialize the algorithm.

V. Experimental Evaluation

In this section we analyze convergence properties of the algorithm and compare our algorithm to K-Means and to the Consistency-Based Detector (CBD) [18] for the problem of motion clustering [1].

Experiments were performed in an urban environment with a car and consist of 30 pairs of laser scans selected from a trajectory of about 2 km. The data reflect typical driving situations such as cars overtaking other cars, crossing by and moving on the opposite lane. For each pair, the scans were taken at 2m to 4m apart which corresponds to the vehicle motion during the data acquisition. Laser points for each pair were manually assigned to different clusters for ground truth purposes. To evaluate how the approach deals with imperfect data associations between laser scans, we ran our algorithm with both the true, manually generated associations between laser points (CRF-T), and the associations computed automatically via CRF-Matching (CRF-M) [13].

Once we obtained a clustering solution, we can compare it with the ground truth assignments. To evaluate the clustering performance, we used the V-measure [15], an external, entropy based, cluster evaluation measure. This measure, V, is the harmonic mean of homogeneity H and completeness C of the cluster assignments. Homogeneity reflects the fact that points in one cluster should belong only to one class and completeness reflects the fact that points in one class should be associated only to one cluster. The V-measure is

[1]CRF-Clustering also estimates the motion for detected objects which is not directly possible with other methods without additional techniques.

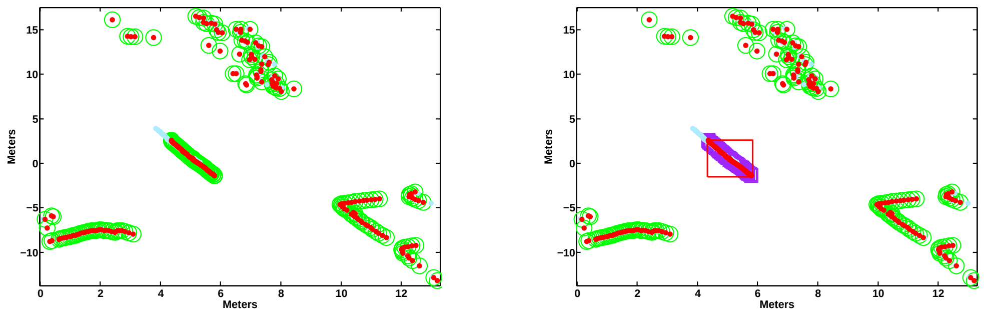


Fig. 4. Example of consistency-based (CBD) results (left) and CRF-Clustering results (right). Red dots indicate the current scan, blue dots the previous one (mostly hidden). The colored markers around dots indicate the cluster assignments. CBD fails to detect the moving object (all points in the same green cluster), where CRF-Clustering doesn't The bounding box on the right image indicates the moving object being correctly detected.

then computed as the harmonic mean of homogeneity and completeness

$$V_\beta = \frac{(1+\beta)HC}{(\beta H) + C} \tag{9}$$

where β is a blending factor. For $\beta > 1$ completeness is weighted more strongly and for $\beta < 1$ homogeneity is weighted more strongly. In our experiment we were interested in both, so we set the blending factor to 1.

A. Convergence Properties

In most of the experiments the algorithm converged between 3 and 7 iterations. One particular case is illustrated in Figure 3. To initialize the algorithm, we segment the scan based on a distance heuristic. The pictures show how this initial segmentation is then refined by CRF-Clustering, while selecting the correct number of clusters. The first three pictures show the likelihood of each laser beam assigned to the clusters (dark red is high likelihood, dark blue is low likelihood). Based on the likelihoods, the last picture shows how the merging procedure described before is able is combine different clusters to obtain the correct solution.

B. Comparison with Consistency-based Detection

In this section we compare CRF-Clustering with the consistency-based detector (CBD) introduced in [18]. The CBD algorithm is a heuristic-based algorithm for detecting moving objects in range data. The main concept behind the algorithm is that static objects are consistent about the *free space* while dynamic objects are not. The major drawback of this algorithm is that it is based on two main assumptions: a good estimate of the robot displacement is available; the object movements are orthogonal to the observed shape. While the first assumption often holds in real situations (use of inertial units, GPS, scan matching), the second is more subtle and can create problems especially in outdoor environments.

Figure 4 shows a typical example in which the second assumption is violated. The robot is approaching an intersection while another car is moving in front of it. We see (left) that CBD is not able to detect the car. That is because most of the car measurements are not classified as dynamic due to the big overlap. On the other hand, CRF-Clustering (right) is able to correctly detect the moving car by clustering laser points according to their motion pattern.

	CBD		**K-Means**		**CRF-T**		**CRF-M**	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
H	0.821	0.279	0.415	0.272	0.983	0.049	0.862	0.052
C	0.893	0.298	0.950	0.065	0.990	0.029	0.903	0.031
V	0.850	0.290	0.537	0.247	0.986	0.039	0.886	0.041

TABLE I
COMPARISON BETWEEN CBD, K-MEANS AND CRF CLUSTERING

Table I shows a numerical comparison between the two techniques. Mean and standard deviation for homogeneity, completeness and V-measure are presented for the different approaches. CRF-Clustering obtains better results in both cases, with true data association (CRF-T), and data association using CRF-Matching (CRF-M).

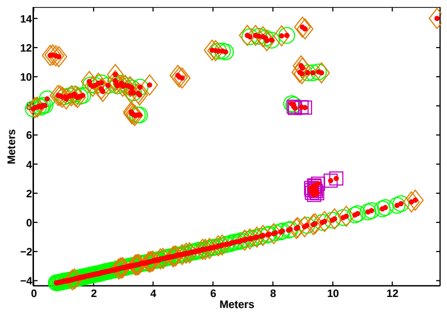
C. Comparison with Modified K-Means

In this section we compare CRF-Clustering with a modified version of the K-Means algorithm. K-Means is a well known and standard algorithm for clustering data points into k partitions. However, K-Means cannot be directly applied to our motion clustering scenario. The problem is that we are clustering points (which are a pair of 2D objects) according to their motion (which is a 3D quantity). The first modification lies in the way the cluster centroids are computed. In our case, the centroids represent the rigid body transformation underlying the object movement (rotation and translation), which is computed according to (3). Once the centroids are obtained, we associate point i to the cluster which minimize

$$\underset{j}{\operatorname{argmin}} \|g_i - (T_j + R_j s_i)\|^2 \tag{10}$$

where (g_i, s_i) is the point pair, T_j and R_j are translation and rotation of the j-th motion cluster.

The main problem of K-Means and similar algorithms is that they do not consider relations between points in the data. More specifically, they assume that points are independent. Discarding this information can lead to very noisy and inhomogeneous clustering results. This is clearly depicted in (5), where we compare the result of K-Means and CRF-Clustering on a typical case. As can be seen, K-Means (left) produces a very noisy result, while the result provided by CRF-Clustering (right) is more accurate and homogeneous. Both algorithms were initialized with the maximum number of dynamic objects (three in our case) and we can see



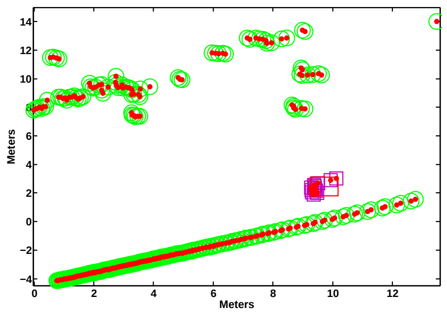


Fig. 5. Modified K-means results (left) and CRF-Clustering results (right). The different symbols indicate the cluster assignment and the red dots indicate the current scan. It can be noticed that the clustering solution found by K-Means is significantly noisy (various different clusters). This is avoided with CRF-Clustering with the use of the neighborhood dependencies. The bounding box on the bottom image indicate the moving object being correctly detected.

that CRF-clustering is able to detect the correct number of clusters.

Finally, Table I shows a numerical comparison between the two techniques on the data collected by our vehicle. Note that K-Means results are based on the manually generated ground truth associations.

VI. Conclusion and Future Works

We have introduced CRF-Clustering, a novel technique for clustering dependant data into homogeneous partitions. Although it is a general clustering algorithm, in this paper we show its ability to detect and classify moving objects in range data. Existing approaches in moving object detection as CBD are mainly based on scan consistency, classifying points as dynamic if they violate the free space of the map. On the contrary, our technique explicitly reasons about the underlying motion of the object, thus being more effective. By using a Conditional Random Field, our approach is able to consider relations between different points in the scans and different properties of the moving objects. Moreover, our algorithm is also able to estimate the underlying motion of the different objects, which can be used as input to a tracking algorithm.

Our experiments show that CRF-Clustering performs better than Consistency-based techniques, specially in situations where the motion of the object is not orthogonal to the observed shape. We also showed that this problem is not trivial from a clustering perspective. Classical algorithms, such as K-Means, fail to provide homogeneous cluster, as they assume data points are independent.

In future works we will investigate various extensions to the CRF-Clustering algorithm. We plan to integrate appearance features for better clustering objects that share the same motion and also the use of other sensors as cameras. Finally, we plan to integrate CRF-Clustering within a Simultaneous Localization and Mapping framework, in order to obtain autonomous navigation and mapping in dynamic environments.

Acknowledgments

The authors would like to thank Dieter Fox for useful discussions. This work is partly supported by the ARC Centres of Excellence programme funded by the Australian Research Council (ARC) and the New South Wales State Government, and the German Research Foundation (DFG) under contract number SFB/TR-8.

References

[1] D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2002.
[2] Y. Bar-Shalom and X.-R. Li. *Multitarget-Multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, 1995.
[3] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards object mapping in dynamic environments with mobile robots. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2002.
[4] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.
[5] Dempster, Laird, and Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
[6] O.R. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley-Interscience, second edition, 2001.
[7] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.
[8] Kurt Koffka. *Principles of Gestalt Psychology*. Lund Humphries, London, 1935.
[9] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2001.
[10] D. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45(3, (Ser. B)), 1989.
[11] L. Montesano, J. Minguez, and L. Montano. Modeling the static and the dynamic parts of the environment to improve sensor-based navigation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.
[12] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., 1988.
[13] F. Ramos, D. Fox, and H. Durrant-Whyte. CRF-matching: Conditional random fields for feature-based scan matching. In *Proc. of Robotics: Science and Systems*, 2007.
[14] D. Rodriguez-Losada and J. Minguez. Improved data association for icp-based scan matching in noisy and dynamic environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2007.
[15] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proc. of Joint Conf. on Emp. Methods in Natural Language Processing*, pages 410–420, 2007.
[16] D. Schulz, W. Burgard, D. Fox, and A. B Cremers. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2001.
[17] Gian Diego Tipaldi. *Looking Inside for Mapping the Outside: Introspective Simultaneous Localization and Mapping*. PhD thesis, University of Roma ”La Sapienza“, 2009.
[18] C. Wang, C. Thorpe, M. Hebert, and H. Thrun, S.and Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *The Int. J. of Robotics Research*, 26(6), June 2007.
[19] D. Wolf and G. Sukhatme. Mobile robot simultaneous localization and mapping in dynamic environments. *Autonomous Robots*, 19:53–65, 2005.
[20] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *Int. J. of Computer Vision*, 13(2):119–152, 1994.