Real-Time Clustering for Long-Term Autonomy

Lionel Ott and Fabio Ramos

Abstract In the future robots will have to operate autonomously for long periods of time. To achieve this they need to be able to learn directly from their environment without human supervision. The use of clustering methods is one possibility to tackle this challenge. Here we present extensions to affinity propagation, a clustering algorithm proposed by Frey and Dueck [5], which makes it suitable for real-time and long-term use in robotics applications. The proposed extension, called meta-point affinity propagation, introduces so called meta-points which increases the performance of the clustering and allows for incremental usage. Additionally we propose a method that enables us to obtain probabilistic cluster assignments from any affinity propagation based clustering method. We show experimental results on the quality and speed of meta-point affinity propagation as well as the probabilistic cluster assignments. Furthermore, we demonstrate how meta-point affinity propagation allows us to process data which is far beyond what affinity propagation could handle.

1 Introduction and Related Work

Our long-term vision is to enable robotic systems to explore and build models of unknown environments over extended periods of time without supervision or prior knowledge. This requires methods that allow the robot to build a model from observations in an unsupervised fashion. Such methods need to be efficient as robots typically have limited resources and the operations need to be performed in a timely

Lionel Ott
Australian Centre for Field Robotics, School of IT, University of Sydney,
e-mail: l.ott@acfr.usyd.edu.au

Fabio Ramos
Australian Centre for Field Robotics, School of IT, University of Sydney,
e-mail: f.ramos@acfr.usyd.edu.au

manner. In addition to creating a model, another important part is the exploitation of the model for higher-level tasks crucial for autonomy such as obstacle avoidance, exploration or scene understanding.

We approach building such environment models from a clustering perspective. Clustering is an unsupervised learning method that groups similar data together into clusters. In our case, the data are the sensory readings of the robot. However, with no prior information about the data, only methods that can infer the number of clusters automatically are suitable.

The contributions presented in this work are two fold. First, we introduce meta-point affinity propagation, a novel method to efficiently cluster large amounts of noisy data. Secondly, we present a method that allows any algorithm based on affinity propagation to obtain probabilistic clustering assignments. We show how these contributions allow us to cluster large numbers of points in a fraction of the time it takes standard affinity propagation. Additionally, we demonstrate how the probabilistic interpretation of cluster assignments can be used to evaluate the quality of clustering results.

1.1 Related Work

The most prominent clustering methods that are capable of inferring the number of clusters from data are latent Dirichlet allocation [2], spectral clustering [10], DB-SCAN [4] and more recently affinity propagation [5]. All of these methods differ in the types of assumptions they make, their complexity and flexibility. Unsupervised learning has been used by Happold et al. [7] in order to learn colour based models which enables them to predict terrain traversability from image data. A different approach with the same goal was proposed by Kim et al. [8] in which they use the experience of a robot as it drives through the environment to learn a model which maps visual appearance to terrain traversability. Another use of clustering is to learn an appearance model in an unsupervised manner. Giguere et al. [6] for example use k-means clustering to learn the model of a coral reef for the purpose of steering a robot such that it remains above the coral reef. The work by Steinberg et al. [13] uses Dirichlet process mixture models to learn models of the benthic habitats present in image data gathered by an AUV. Such methods are not limited to image data as shown by Modayil and Kuipers [9] who learn object models from laser scan data using a simple clustering method. A different method that learns models from 3D point clouds is presented by Ruhnke et al. [12] who employ spectral clustering to cluster models based on their consistency.

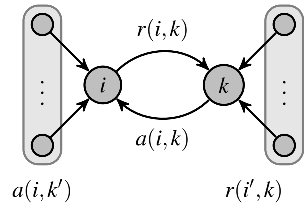


Fig. 1: Messages exchanged by affinity propagation between nodes in each iteration. Both messages take into account the accumulated values of the other message at the given node.

2 Approach

We cast the task of building a model of the environment as a clustering problem. We use affinity propagation [5], a state-of-the-art clustering method, to build the model of the environment. Affinity propagation has many advantages for our purpose. First, there is no need to define the number of clusters a priori as they are determined from the data itself. Second, the only inputs required are the similarity values between data points which can be any sensible value in the context of the application, and is not required to be a metric. These similarity values are then used by affinity propagation to compute the clustering solution by iteratively computing two messages, availability and responsibility. Availability $a(i,k)$ is the message sent from point k to point i and encodes how good of an exemplar k would be for i based on evidence available to point k. Responsibility $r(i,k)$ sent from point i to k on the other hand encodes how suited point k is as an exemplar for point i given the information available in i. After initialising all messages to 0 they are computed iteratively until convergence is achieved. The actual equations used to compute the messages are shown below:

$$r(i,k) = s(i,k) - \max_{k' \text{s.t.} k' \neq k} \left(a(i,k') + s(i,k')\right) \tag{1}$$

$$a(i,k) = \min\left(0, r(k,k) + \sum_{i' \text{s.t.} i' \notin \{i,k\}} \max\left(0, r(i',k)\right)\right) \tag{2}$$

$$a(k,k) = \sum_{i' \text{s.t.} i' \neq k} \max\left(0, r(i',k)\right) . \tag{3}$$

Figure 1 shows how these two messages interact with each other. One can see that both messages are computed using the values of the other messages accumulated at the node.

In this work, we use cameras to perceive the robot's environment and thus need a compact way to represent the visual appearance of observations. To this end we split the images observed by the robot into small rectangular patches. For a 640×480 image the patches are typically 80×60 in size. From these, we extract HSV colour space histograms and histograms of local binary patterns [11], thus capturing both colour and texture information. The similarity between features obtained from observations in this way is computed as the sum of the histogram similarities, i.e.:

$$sim(H_1, H_2) = -dist(H_1^{\text{colour}}, H_2^{\text{colour}}) - dist(H_1^{\text{texture}}, H_2^{\text{texture}}) , \tag{4}$$

where H_1 and H_2 are the histograms of the colour and texture information for each of the image patches. The distance *dist* between two histograms is computed using the Bhattacharyya distance of two histograms.

2.1 Meta-Point Affinity Propagation

While standard affinity propagation produces good results, it is too slow to process thousands of data points in a few seconds. We therefore propose a method called meta-point affinity propagation which is inspired by ideas presented in Cao et al. [3]. The main idea is that data points which are close in feature space can be grouped together and replaced by a single meta-point. In robotics, similar observations occur frequently for example multiple observations made from a similar pose. By replacing such redundant observations with a single aggregated one, we effectively reduce the number of points involved in the computation of affinity propagation.

A meta-point $\mathbf{P}_i$ stores the following information:

$$\mathbf{P}_i = \{\text{count}, \text{mean}, \text{exemplar}, \text{last-update}\} , \tag{5}$$

with the fields having the following meaning:

$\mathbf{P}_i$.count	number of points represented by the meta-point
$\mathbf{P}_i$.mean	the mean value of all represented data points
$\mathbf{P}_i$.exemplar	representative raw data point for this meta-point
$\mathbf{P}_i$.last-update	time the meta-point has been updated last

Besides the immediate effect of reducing the computational burden the concept of meta-points has two additional benefits:

- the number of meta-points is dependant on the size of the feature space;
- random observations can be dealt with in a straight forward way.

The first point is a direct consequence of the usage of meta-points instead of raw data points. If a robot moves in a static environment all observations will be mapped to one of the meta-points after a while and thus no new meta-points will be created. The second point requires us to distinguish between two types of meta-points: Cluster-points that represent the points used for clustering, and noise-points which are ignored during the clustering. A meta-point is considered a cluster-point once it represents enough raw data points, otherwise it is considered a noise-point. This allows us to discard points generated from random observations such as spurious readings from a laser scanner. Put differently we can detect and ignore outliers in our observations.

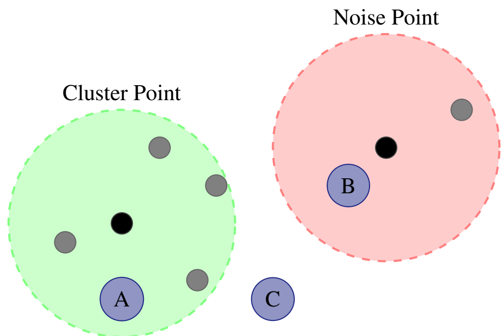


Fig. 2: Visualisation of meta-points and cases that can occur when adding a new data point. *A* is merged into the cluster-point while *B* is merged into the noise-point. Finally, *C* is used to create an entirely new meta-point.

The most important part of meta-point affinity propagation is the handling of new observations. The pseudo code in Algorithm 1 shows the steps performed in order to add a point p into either the set of cluster-points $\mathbf{P}$ or the set of noise-points $\mathbf{N}$. Figure 2 shows the possible cases described above and in the pseudo-code. We keep these two sets separate for performance reasons. A new data point is added to an existing data point, either cluster-point or noise-point, if the meta-point is similar enough to the data point. Otherwise a new meta-point is created from the new raw data point. In case that the data point was added to a noise-point and this one now represents enough points to be considered a cluster-point is moved to the set of cluster points $\mathbf{P}$.

Computing the actual clustering result is then performed using standard affinity propagation using the cluster-point data. The two parameters required by meta-point affinity propagation are, $\theta_{\text{min-points}}$ the minimal number of points required for a meta-point to be considered during the clustering and $\theta_{\text{similarity}}$ the maximal difference in similarity between a meta-point and a new point for it to be able to be considered part of that meta-point. The similarity threshold is tied to the range of values the chosen similarity measure can take on. The minimum number of points is related to the rate at which observations are made. If the value is too low many points that can be considered noise will be added and if it is too high actual clusters that appear only rarely may not be added. In order to prevent noise-points to turn into meta-points by accumulating over long periods of time one can also purge noise-points that have not been observed for a set period of time. In the extreme case where both parameters are set to zero we end up with the original affinity propagation algorithm again.

This form of merging data points obviously assumes that small changes in the feature space distance result in no noticeable change of the object class to be clustered. Additionally the handling of noise only addresses noise which results from random measurements or one off sensing failures. It does not detect or handle com-

```
ADD-DATA-POINT(p)
nn = NEAREST-NEIGHBOUR(P, p)
if DIST(nn, p) < θ_similarity
    UPDATE-META-POINT(nn, p)
else
    nn = NEAREST-NEIGHBOUR(N, p)
    if DIST(nn, p) < θ_similarity
        UPDATE-METAPOINT(nn, p)
        if nn.count < θ_min-points
            P = P ∪ nn
            N = N \ nn
    else
        noise = CREATE-META-POINT(p)
        N = N ∪ noise
```

Algorithm 1: Pseudo code detailing the steps performed by meta-point affinity propagation when a new data point is added. **P** is the set of cluster-points, **N** the set of noise meta-points and θ the parameters.

plete failure of a sensor or systematic noise, as these produce consistent and continuous observations.

2.2 Probabilistic Cluster Assignments

Affinity propagation performs hard cluster assignments, i.e. each data point is assigned to exactly one cluster. Often times, however, assignments are not this clear-cut. Furthermore probabilistic methods are widely used in robotics since they enable us to deal with uncertainties of representations in a principled way. Thus a probabilistic interpretation of the clustering would be highly beneficial. As it turns out we can derive such an interpretation by analysing the internal representation of affinity propagation in the following way.

We start by finding the set of exemplars E in the typical way of affinity propagation, that is by selecting set of indices k for which:

$$e_k \in E : a(k,k) + r(k,k) > 0\,, \tag{6}$$

holds, i.e. all points for which the value of the self-availability and self-responsibility is greater then zero. The probability of an observation i to belong to cluster k is then defined as

$$p(i = k) = \frac{1}{Z} f(a(i,k) + r(i,k))\,, \tag{7}$$

where

$$Z = \sum_{k \in E} f(a(i,k) + r(i,k)) \tag{8}$$

is the normalization factor and f is a function that maps its inputs to the range $[0,1]$. In our case we use the logistic function:

$$f(x) = \frac{1}{1+e^{-x}}\,, \tag{9}$$

as it covers the values typically taken on by affinity propagation well. From this probability distribution we can obviously recover the hard assignment affinity propagation makes as:

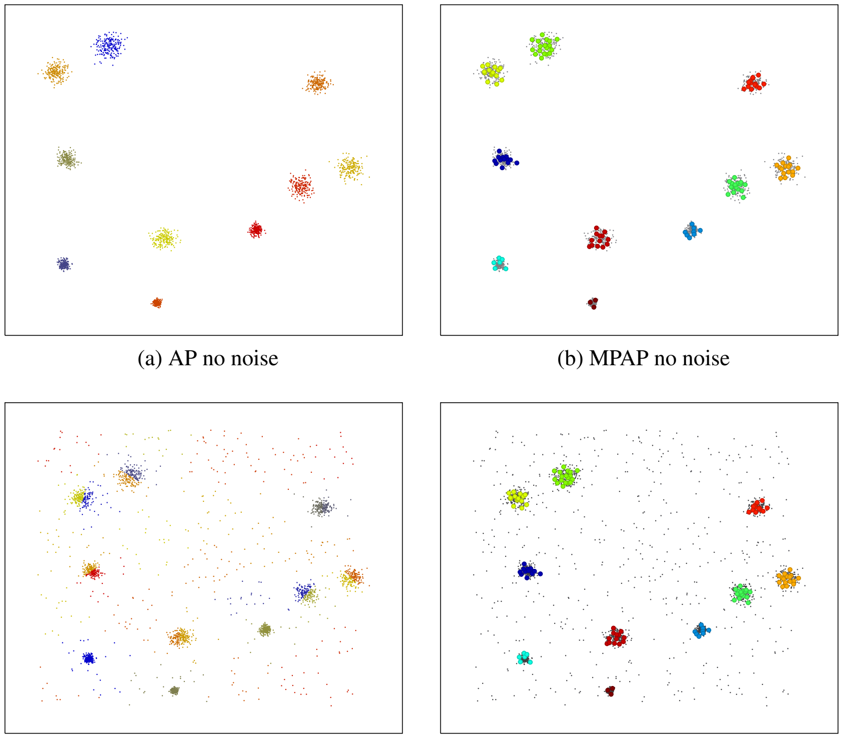
$$\operatorname*{argmax}_k p(i = k) \tag{10}$$

However, we also gain the ability to reason about an observation in terms of its fit with respect to the current clustering. From the entropy of the assignment, obtained as:

$$H(i) = -\sum_{k=1}^{n} p(i = k) \log p(i = k)\,, \tag{11}$$

we can tell how well the clustering can explain the observation. A small entropy value indicates a well explained data point, i.e. a peaked distribution, where as a large value indicates an observation that can be explained similarly well by multiple clusters.

In addition to the evaluation of a single observation we can extend this to the entire clustering solution in order to obtain an overall quality measurement. A measure of the quality of the clustering solution is interesting as for humans it is easy to tell if the result obtained by clustering is meaningful. However, a robot lacks this intuition and thus a way to quantify the quality of clustering results is of great importance. The evaluation of clustering results has been the focus of intense research and there are different methods to obtain such a measure, see [1] for an overview. However, most of the proposed methods are intended to assign a score to a clustering solution in order to compare different clustering methods against each other. As such they require a reference clustering solution as ground-truth. In robotics, however, obtaining ground-truth is often hard, if not impossible, and in our case no reference clustering exists. For this reason methods that rely on a reference clustering are not suitable for our application. A possible way to evaluate the clustering solution with the information we have access to is to compute an average entropy. This indicates how well all individual points clustered can be represented by the end result. Here again a low value indicates a good solution.



(a) AP no noise (b) MPAP no noise

(c) AP with noise (d) MPAP with noise

Fig. 3: (a), (b) Exemplary results of affinity propagation and meta point affinity propagation on data without noise. (c), (d) Exemplary results of affinity propagation and meta-point affinity propagation with uniform noise. The small circles indicate the meta-points found by meta-point affinity propagation. The colouring indicate points that have been assigned to the same cluster.

3 Experiments

3.1 Clustering Results and Speed

In a first set of experiments we compare the clustering results obtained by affinity propagation and meta-point affinity propagation on synthetic data, as shown in Figure 3. We use this to easily show the differences between the two algorithm. In the case without noise we can see that both methods find a good clustering solution, as indicated by the coloured points. The difference between the two is that affinity propagation has to cluster all the data points, whereas meta-point affinity propagation only has to cluster the meta-points, indicated by the bigger circles. The large difference in speed this makes can be seen in Table 1.

As mentioned in Section 2.1, an additional advantage of meta-point affinity propagation over affinity propagation is the ability to deal with random observations. For the synthetic data we simulate such random observations by adding data points from a uniform distribution. Figure 3c and Figure 3d show typical results for both methods under these conditions. One can see that affinity propagation creates a larger number of clusters compared to the result obtained on the same data without noise. This is due to the fact that affinity propagation has to assign each data point to a cluster and cannot consider some observations as noise and ignore them. This may in some cases, depending on the chosen parameters, result in clusters to be split as can be seen in Figure 3c. Meta-point affinity propagation on the other hand first builds meta-points which allows the method to reject points it considers to be noise and then the clustering has only to compute the solution for data without or reduced amount of noise. Comparing the meta-points for the data with and without noise we can see that in both cases they cover the actual clusters. Consequently, as far as meta-point affinity propagation is concerned, there is no noise in the data to be clustered.

From the plots of the synthetic data it is easily visible how meta point affinity propagation represents the original data with fewer data points. This allows meta point affinity propagation to be significantly faster then affinity propagation, as the numbers in Table 1 show. As affinity propagation has quadratic runtime the gaps between the two methods will keep increasing if more points from the same underlying model are added as meta-point affinity propagation will only update the meta-point statistics while affinity propagation has to handle entirely new points.

Figure 4 shows the exemplars obtained when clustering 1200 image patches that represent tree, grass, brick wall, asphalt, red concrete and wood chips. One can see that both methods find the same types of clusters with the big difference being the number of points involved in the clustering and the resulting speed which is shown in Table 1. The timing values include the entire processing of the data, i.e. feature extraction and management of meta-points. The reduction in data points by meta-point affinity propagation is quite drastic as only around 10% of the original data is retained while still producing the same clustering result.

Finally, we used meta-point affinity propagation to batch-process images captured while the robot was moving through the environment for 15 to 30 minutes. The clustering of these images requires several thousands of data points to be handled. The "Large-Scale Data" section in Table 1 shows the exact numbers. Standard affinity propagation can't handle this amount of data in a reasonable amount of time. However, using meta-point affinity propagation we are able to reduce these numbers to manageable sizes, requiring just over one percent of the original amount of data to be retained. The durations shown are obtained by processing large batches of images to meta-point affinity propagation at once and then running the clustering. This is repeated until all images have been added. The final exemplars obtained in this way for the outdoor dataset is shown in Figure 5.

Fig. 4: Exemplars found by both methods for a set of 1200 patches from an outdoor data set. From left to right we can see: brick wall, asphalt, tree, grass, wood chips and red concrete.

Fig. 5: The exemplars obtained from batch-processing around 900 images collected in an outdoor environment using meta-point affinity propagation. The exemplars are not as unique as in Figure 4, however, the data was not pre-processed either.

The results shown here from both synthetic and real images shows that meta-point affinity propagation obtains results that are comparable with affinity propagation but at a fraction of the computational cost. The added robustness of meta-point affinity propagation to noise makes this new method very appealing for use in robotics.

3.2 Probabilistic Cluster Assignments

To show the performance of our probabilistic cluster assignment we build a model out of the previously used 1200 image patches representing grass, trees, wood chips, asphalt, brick wall and red concrete. The exemplars obtained from the clustering are shown in Figure 4. We can see that for images containing the appearance of only a single cluster, as shown in Figure 6 the probability distribution is peaked around the corresponding cluster. When we look at the more interesting cases in Figure 7 where there is more then one cluster represented, as is the case in the top two images, we can see that they contribute the two largest peaks in the distribution. Finally, if we look at the last image in Figure 7 which obviously does not belong to any of the clusters we can see that the overall distribution is rather flat compared to the other cases. Compared to the hard assignment of affinity propagation which would have

Synthetic Data				
Method	Clusters	No. raw points	No. clustered points	Duration (s)
AP no noise	10	2000	2000	60.61
AP with noise	35	2500	2500	63.43
MPAP no noise	10	2000	121	0.08
MPAP with noise	10	2500	122	0.08
Outdoor Data				
Method	Clusters	No. raw points	No. clustered points	Duration (s)
AP	6	1200	1200	8.79
MPAP	6	1200	79	1.84
Large-Scale Data				
Dataset	Clusters	No. raw points	No. clustered points	Duration (s)
Outdoor	10	31464	594	185
Indoor	10	48600	626	307

Table 1: Results for different clustering tasks. Synthetic data shows the results of the 2D example while indoor and outdoor data show the numbers for the corresponding data sets. The number of clusters obtained as well as the number of actual data points clustered with the total run-time is shown.

assigned this image to the “red concrete” class the probability distribution informs us that we can not trust the cluster assignment much.

4 Conclusion

In this paper we presented an extension to affinity propagation called affinity propagation, which allows us to cluster data in real-time and incrementally. Furthermore we proposed a generic way to extract probabilistic cluster assignments for affinity propagation based methods. In experiments we show how meta-point affinity propagation obtains results similar to affinity propagation, however, at a much lower computational burden. We also show how the probabilistic cluster assignments can help to evaluate the clustering of a single observation or the entire clustering result. This combination of incremental real-time clustering with probabilistic assignments will allow us to build meaningful models of the environment a robot operates in. Such models will adapt to changes in the environment and can also be used for example to guide the robot in the exploration of its surroundings.

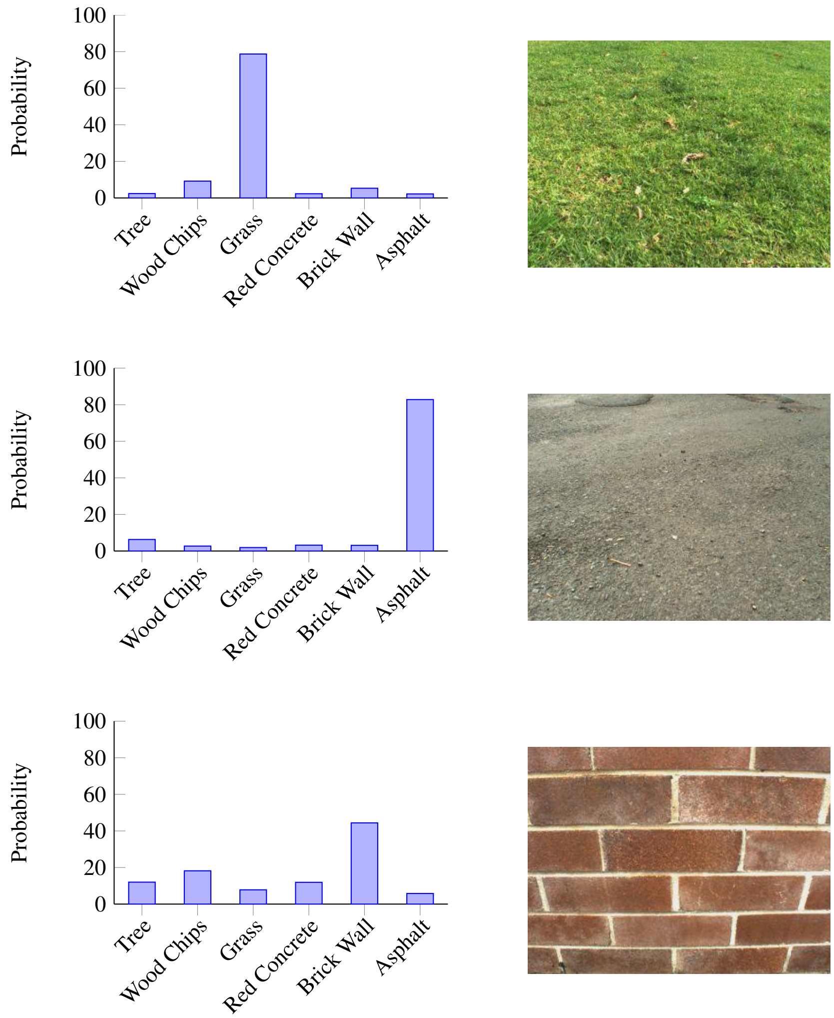


Fig. 6: Examples of probability distributions for single observations of uniform appearance. From top to bottom we have grass, asphalt and brick wall.

References

[1] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of Extrinsic Clustering Evaluation Metrics based on Formal Constraints. *Information Retrieval*, 2008.

[2] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003.

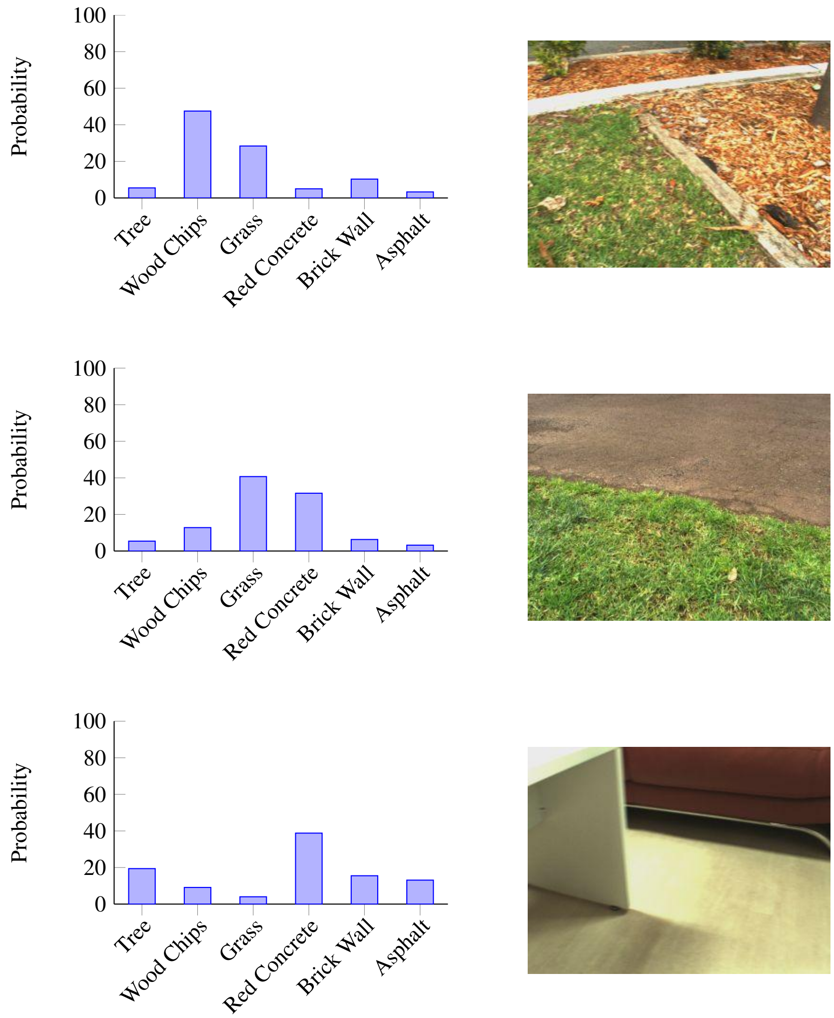


Fig. 7: Examples of probability distributions of single observations with mixed or unknown appearance. From top to bottom we have a scene with grass and wood chips, grass and red concrete and finally an indoor seating area which is not covered by the clustering.

[3] F. Cao, M. Ester, W. Qian, and A. Zhou. Density-Based Clustering over an Evolving Data Stream with Noise. In *SIAM International Conference on Data Mining*, 2006.

[4] M. Ester, H. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. of the International Conference on Knowldge Discovery and Data Mining*, 1996.

[5] B.J. Frey and D. Dueck. Clustering by Passing Messages Between Data Points. *Science*, 2007.

[6] P. Giguere, G. Dudek, C. Prahacs, N. Plamondon, and K. Turgeon. Unsupervised Learning of Terrain Appearance for Automated Coral Reef Exploration. In *Proc. of the Canadian Conference on Computer and Robot Vision*, 2009.

[7] M. Happold, M. Ollis, and N. Johnson. Enhancing Supervised Terrain Classification with Predictive Unsupervised Learning. In *Proc. of Robotics: Science and Systems (RSS)*, 2006.

[8] D. Kim, J. Sun, J.M. Rehg, and A.F. Bobick. Traversability Classification using Unsupervised On-line Visual Learning for Outdoor Robot Navigation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2006.

[9] J. Modayil and B. Kuipers. Bootstrap Learning for Object Discovery. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[10] A. Ng, M. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. In *Proc. of Advances in Neural Information Processing Systems*, 2001.

[11] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[12] M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard. *Unsupervised Learning of 3D Object Models from Partial Views*. 2009.

[13] D. Steinberg, O. Pizarro, S. Williams, and M. Jakuba. Dirichlet process mixture models for autonomous habitat classification. In *OCEANS 2010 IEEE*, 2010.