Noname manuscript No.
(will be inserted by the editor)

PLISS: Labeling Places Using Online Changepoint Detection

Ananth Ranganathan

the date of receipt and acceptance should be inserted later

Abstract A shared vocabulary between humans and robots for describing spatial concepts is essential for effective human robot interaction. Towards this goal, we present a novel technique for place categorization from visual cues called PLISS (Place Labeling through Image Sequence Segmentation). PLISS is different from existing place categorization systems in two major ways - it inherently works on video and image streams rather than single images, and it can detect "unknown" place labels, i.e. place categories that it does not know about. PLISS uses changepoint detection to temporally segment image sequences which are subsequently labeled. Changepoint detection and labeling are performed inside a systematic probabilistic framework. Unknown place labels are detected by using a probabilistic classifier and keeping track of its label uncertainty. We present experiments and comparisons on the large and extensive VPC dataset. We also demonstrate results using models learned from images downloaded from Google's image search.

1 Introduction

Place categorization is the problem of tagging places with semantically meaningful labels. As opposed to place recognition, which deals with the task of consistently identifying a specific place every time it is visited and is useful in robotic mapping, categorization deals with higher-level knowledge and semantic mapping. Some examples of categories are "Kitchen", "corridor", "Library", and "Fast food joint". As is immediately evident, place categorization is dependent on the task at hand and the resolution of semantic knowledge required since there can be multiple labels at different spatial levels applicable to the same place. Place categorization is essential in order for a robot or an intelligent agent to semantically understand places in a manner similar to that done by people. It is also useful, if not a requirement, for semantic mapping and can provide contextual cues for many problems in computer vision. For instance, the label of a place places a strong prior on the types of objects found there [41].

Honda Research Institute USA, Inc

Most existing place recognition systems are classifier based, and assume a finite set of place labels which are learned offline from supervised training data. The system may learn a one-vs-all classifier or a separate binary classifier for each label. The classifier then categorizes input images sequentially during runtime. The advantage of these systems is simplicity and the availability of off-the-shelf components such as Support Vector Machines (SVMs) [37].

However, systems based on straight-forward application of classifiers also have a few significant disadvantages. Firstly, classification is performed independently for each input image even though the robot is physically constrained to only move from one place to another infrequently so that there is a strong continuity bias in the place label. Discriminative classifiers cannot, in most cases, take advantage of this information though it can be modeled generatively [53]. Instead, output label continuity is externally enforced through methods such as Bayesian filtering [46]. Secondly, since classifiers, particularly discriminative ones, are constrained to only provide labels from the known label set, these systems cannot flag a place category that has not been previously encountered. This inability to signal an "unknown" label results in incorrect output and poor performance.

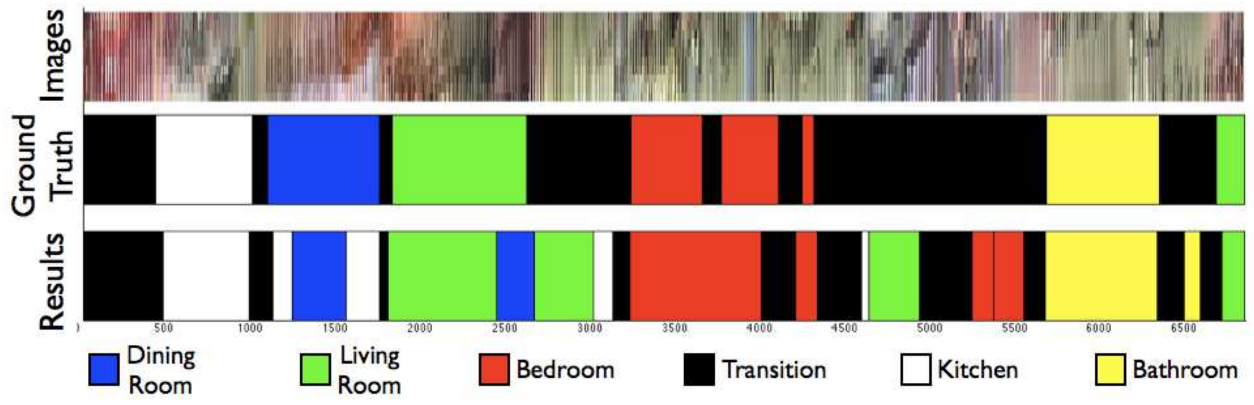


Fig. 1: Maximum likelihood output labeling for the 1st floor image sequence from Home 3 of the VPC dataset, which is one of the more difficult ones and contains 6839 images. Thumbnails (in the shape of thin "strips") of original images, groundtruth labeling and the PLISS result are shown. Note that it is possible to spot many of the changepoints from just the high-level image characteristics visible here but not all of them.

In this paper, we present a novel system for place categorization called PLISS, which stands for Place Labeling through Image Sequence Segmentation. PLISS is based on the observation that the place label does not change at every measurement and works by temporally segmenting image sequences into segments corresponding to different place categories. We use changepoint detection for temporal segmentation. Changepoint detection, as it's name suggests, is the problem of detecting instants in time when the measurement model characteristics undergo significant changes. If the modeling is performed in such a manner as to make these changes correspond to the robot entering and exiting places, we can obtain a segmentation where each segment corresponds to a place category. This provides a natural way of approaching the place categorization problem and yields smooth output without the need for additional filtering of the result.

It may be argued that a weakness of this method is it's dependence on a good temporal segmentation. We address this issue by performing robust segmentation through changepoint detection within a Bayesian probabilistic framework which systematically takes all possible segmentations into account at each time step and computes a posterior distribution over temporal segmentations. This is done in an online manner so as to be of practical use, i.e. we do not need the complete image sequence for the changepoint inference. Moreover, unlike most cases involving Bayesian inference over complex spaces, this inference can be performed exactly and efficiently to obtain near real-time performance. We also present a particle filter approximation to the exact algorithm that further improves efficiency without compromising on performance. The temporal segments are labeled using place models learned offline using training data. We use a Gaussian Process classifier (GPC) [33] for this purpose since this also provides classification uncertainty at the query point. We use this uncertainty to flag the case of unknown labels in a systematic manner. We demonstrate that this provides a scalable technique for handling the unknown label problem.

Hence, PLISS elegantly overcomes the disadvantages of discriminative classifier based systems mentioned above, i.e.

1. PLISS is temporally consistent: Since PLISS is based on labeling segments of image sequences obtained using changepoint detection, it is inherently temporally consistent without the need for additional filtering or smoothing methods. Further, the changepoint detection itself performed in a systematic manner like than in an ad-hoc manner.
2. PLISS handles the unknown label problem: PLISS is not constrained to classify the input into the predetermined place category set and can detect scenarios where the place category is unknown

While a few existing generative methods also have the two advantages mentioned above, PLISS is also computationally efficient and can operate very close to real time in practical scenarios, as we demonstrate in our experiments.

We model images using histograms of dense SIFT features. The image histograms are used as measurements in the changepoint detection procedure wherein we use the Multivariate Polya distribution as the measurement model. The multivariate Polya distribution, also known as the Dirichlet Compound Multinomial (DCM) model [25], is a bag-of-words model previously used for document modeling. It captures burstiness of the data, i.e. it models the observation that if a word occurs once in a document, it usually occurs repeatedly. Unfortunately, the DCM model cannot be updated incrementally, and hence we provide an efficient Gaussian approximation based on projecting the high dimensional DCM model into a low dimensional space. The image histograms are also used to learn the GPC place models.

We present extensive experiments for validating PLISS on the Visual Place Categorization (VPC) dataset [46], a large and challenging dataset collected from multiple home environments. The labeling result from PLISS on one of the sequences of the VPC dataset is shown in Figure 1. We also present results using place models learned from images obtained from Google's image search using the place label as the keyword, the advantage being that no knowledge of the operational environment is necessary during training. This problem represents a higher level of difficulty than learning place models from similar environments, and provides a big step towards the practical use of place categorization. We present comparisons against other algorithms and runtime results for PLISS.

The major difference between this work and our prior work on PLISS [31] is that we no longer use Multivariate Polya models for both changepoint detection and place modeling. Instead, place modeling is performed using a Gaussian Process classifier (GPC) and we utilize the uncertainty estimate provided by the GPC for detecting the unknown label case. The advantage is increased classification performance and scalability with number of place labels. This is so because the Multivariate Polya model is more suited for clustering rather than classification and does not generalize well when class overlap is present. In addition, our new technique for detecting the unknown label case is much simpler, faster, and scales better than statistical hypothesis testing based on the Multivariate Polya model, which we used previously. We also present a more efficient approximation to the changepoint detection procedure based on projecting down the high dimensional Multivariate Polya measurement model to a low dimensional subspace. Finally, we present significantly more experiments and comparisons on challenging datasets, including results from learning place models on images obtained from web searches.

2 Related Work

Place categorization has received extensive attention both in robotics and computer vision. In robotics, place recognition and classification fall under the more general areas of spatial reasoning [20] and semantic mapping for robots [49]. They also facilitate human-robot communication [40]. Early work on place recognition was confined to recognizing specific places rather than categories. Typical approaches include matching SIFT features across images [51] or matching other derived measures of distinctiveness for places such as Fourier signatures [24], subspace representations [15], and color histograms [43]. These methods have the disadvantage of not being able to generalize and also need to use omnidirectional cameras to achieve invariance to perspective. Approaches that generalize well usually learn classifiers, involving SVMs or boosting, for each place from labeled data [35,39]. A very recent system by Pronobis et al. [30] focusses on merging cues from different sensing modalities, the output for each of which is obtained from individually trained SVMs. In contrast to PLISS, these and similar discriminative classifier-based approaches cannot detect or learn unknown places and place categories.

Generative methods, of which ours is an instance, can in principle detect unknown categories using outlier detection. Outlier detection has been used in [53], where it is called a bail-out test. Our previous work [31], which used hypothesis testing for detecting new categories, is similar in flavor to that work. However, in the current paper, we use Gaussian Process classifiers for detecting outliers thus eliminating the need for hypothesis testing.

While so far we have discussed only the use of filtering as a way to impose the temporal continuity constraint, a number of other two-tier methods are possible. For instance, in Posner et. al [54], a clustering method is used to create these constraints, which are not only temporal in this case but can also link temporally non-sequential frames. Similarly, segmenting a space into regions based on domain-specific cues such as doors [30], can also considered to be a domain specific version of changepoint detection.

Place classification methods based on laser and sonar range scans also exist. Kuipers and Beeson [19] apply a clustering algorithm to the laser scans to identify distinctive places while Mozos et al. combine features extracted from laser scans and camera images[26] to learn boosted classifiers for place categories. Posner et. al [53] use a Chow-Liu tree model for modeling individual patches. The individual patch likelihoods are integrated within a spatio-temporal MRF on image superpixels.

In computer vision, place classification is also known as scene classification and image categorization. Two broad methods can be discerned - those that model local features [2], objects[32], and distinctive parts of images[38], and those that extract global representations of images and learn from them. The latter is closer to this work and hence, merits closer attention.

Spatial Pyramid Matching [21] introduced the use of histograms of dense SIFT features computed at various spatial scales for representing images. We utilize the same image representation in this work. These histograms are used to learn SVM classifiers which incorporate the histogram intersection kernel [17]. Wu and Rehg [47] model images using quantized histograms of census transform values. They term this the CENTRIST representation and use these to learn SVM classifiers for place categories in their Visual Place Categorization (VPC) system. Other global methods include Bosch et al. [3], who learn a random forest classifier for identifying place categories. The input to the classifier is dense SIFT features. Oliva and Torralba [28] introduced the gist of an image as a global feature for place categorization. Gists are essentially histograms of image gradients computed across the entire image. Fei-fei and Perona [14] introduced a bag-of-words Bayesian generative model based on Probabilistic Latent Semantic Analysis (PLSA) for recognizing scene categories. However, the SVM-based systems mentioned above have since outperformed this system.

Place categorization, in the form of image classification, is also an essential component of content-based image retrieval. However, due to large image databases and the need for fast retrieval, techniques thus far have been limited to color and texture correlograms, and similar simpler features [9]. Many hashing techniques such as Semantic hashing [36] and Spectral hashing [44] have been used for this purpose though they are not directly applicable to our scenario since supervised learning for place labels is not possible in these algorithms.

Changepoint detection has a long history in statistics and the best-known technique is the CUSUM detector [29] which involves modeling the time series as piecewise segments of Gaussian mean with noise. Closer to our application is segmented regression [11]. In our exposition, we closely follow the more general method of [1] which is applicable to conjugate-exponential models. The particle filter algorithm we have presented is also similar to [13] though a version that does not involve Rao-Blackwellization (and hence is less efficient) was presented in [7] first. Applications of changepoint detection in computer vision range from video segmentation [50] to robust visual tracking [42] to visual scene analysis [5], though this is the first application to place categorization to our knowledge.

3 Problem Setup

We formulate the place recognition problem as follows. We are given a measurement stream with measurements at some (possibly changing) intervals, which we will also call as timesteps. For each measurement, we are required to come up with a label corresponding to the

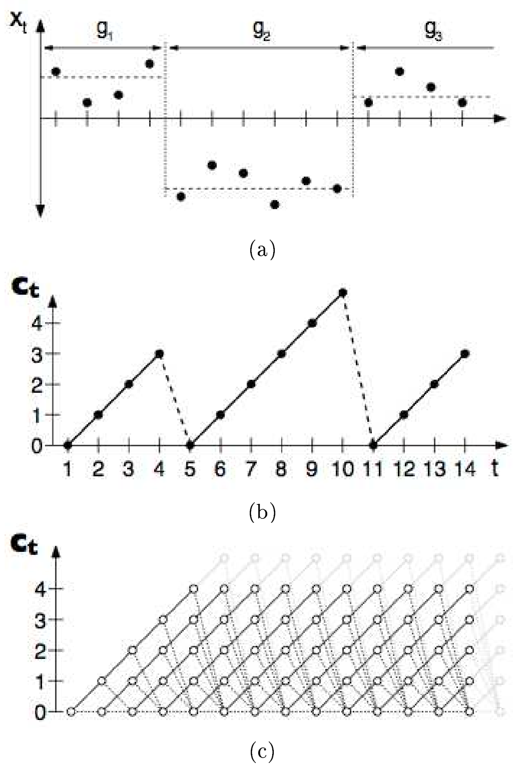
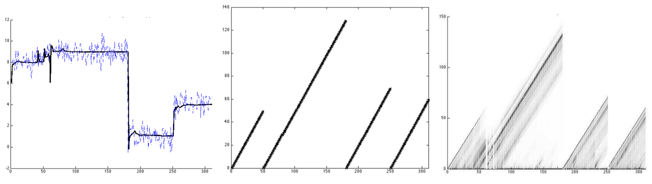


Fig. 2: An explanation of how the changepoint inference is parametrized: (a) Hypothetical data divided into 3 segments (b) The value of the variable length c_t used in our inference as a function of time. c_t is the length of the current segment and resets to 0 when there is a changepoint. (c) The trellis showing all the possibilities over which inference is performed. Solid lines indicate an increase in the length of the current segment (the case of no changepoint) while dotted lines indicate the possibility of a changepoint at that time instant. Figures taken from [1]. The horizontal axis denotes time in all three figures.

type of the place. The place types are given in the form of L place models $M_1, M_2, \ldots, M_L$. We are also required to say if the measurement does not correspond to any of these labels.

We approach the problem by noting that the place label remains the same for periods of time when a robot is moving inside a particular place. It only changes sporadically when the robot travels into the next place. Thus, the measurement stream can be segmented into pieces corresponding to places, i.e. measurements in each segment can be assumed to have been generated by the corresponding place model.

We assume that a sequence of data $y_1, y_2, \ldots, y_t$ can be segmented into non-overlapping but adjacent segments. The boundaries between these segments are the changepoints. We deal with the model-based changepoint scenario here, wherein the form of the probability distribution in each segment remains the same and only the parameter value changes. We assume that the data are i.i.d within each segment and denote by c_t the



(a) (b) (c)

Fig. 3: Changepoint detection: (a) Univariate Gaussian input data (blue/dashed) with segment means shown in black (b) Groundtruth for changepoints shown as length of segments, which is the variable c_t used in our inference. (c) Output of changepoint detection which is probability distribution on c_t. Darker shades mean higher probabilities and the representation is simply a shaded version of the trellis in Figure 2(c). The horizontal axis in all the figures denotes time while the vertical axis in (b) and (c) denotes current segment length. The corresponding axis in (a) refers to magnitude of the measurements.

length of the segment at timestep (or index) t. Note that c_t is also the timesteps since the last changepoint. If the current timestep is a changepoint then $c_t = 0$ and if no changepoints have occurred yet then $c_t = t$. While computing the distribution over c_t, we have to consider all the cases from $c_t = 0$ (changepoint at current timestep), $c_t = 1$ (changepoint at $t-1$), and so on up to $c_t = t$ (no changepoint so far). A illustrative explanation of the c_t variable is shown in Figure 2. A sample problem setup and output for changepoint detection in the univariate Gaussian case in shown in Figure 3.

We denote the place label at timestep t as x_t^c. The place label is indexed by the current segment since the whole segment has a single label. However, this label is updated with each measurement, and hence the time index t is also used. The probability distribution over x_t^c is taken to be a discrete distribution of size L, one for each of the place models. The case where the place label takes none of these values is detected separately.

We need to compute the joint posterior on c_t and x_t^c given the data, $p(c_t, x_t^c | y_{1:t})$, where $y_{1:t}$ denotes all the data from timestep 1 to timestep t. The posterior can be factored as

$$p(c_t, x_t^c | y_{1:t}) = p(c_t | y_{1:t}) p(x_t^c | c_t, y_{1:t}) \tag{1}$$

The first term is the posterior over the segment length while the second term is the conditional posterior on the place label given the segment length. Note that the posterior over segment length is equivalent to inferring the changepoints since $c_t = 0$ implies a changepoint. Inference for the place label need not be performed at each time step, especially if the segment length is large, i.e. $c_t > C$ for some large C so that label inference for the segment is likely to have stabilized. However, since the label posterior is not guaranteed to remain unchanged even if this is the case, we recommend periodically computing it even at this stage.

We address changepoint detection, i.e. computing $p(c_t | y_{1:t})$ in the next section. Computation of the place label posterior $p(x_t^c | c_t, y_{1:t})$ is explained in Section 6.

4 Model-based Online changepoint Detection

In the following exposition, we closely follow [1] and [13], both of which state essentially similar algorithms but with different state representations. We represent the likelihood of the data in segment c_t as $p(y_t | \xi_t^c)$ where ξ_t^c is a parameter set. The data inside each segment are assumed to be i.i.d and the parameters are assumed i.i.d according to a prior parameter distribution.

The changepoint posterior from (1) can be expanded using Bayes law as

$$p(c_t | y_{1:t}) \propto p(y_t | c_t, y_{1:t-1}) p(c_t | y_{1:t-1}) \tag{2}$$

The first term is the data likelihood while the second term can be further expanded by marginalizing over the segment length at the previous timestep to yield a recursive formulation for c_t

$$p(c_t | y_{1:t-1}) = \sum_{c_{t-1}} p(c_t | c_{t-1}) p(c_{t-1} | y_{1:t-1}) \tag{3}$$

where $p(c_t | c_{t-1})$ is the transition probability, $p(c_{t-1} | y_{1:t-1})$ is the posterior from the previous step, and we have made use of the fact that $c_1, c_2, \ldots, c_t$ form a Markov chain.

4.1 The Transition Probability

For characterizing the transition probability $p(c_t | c_{t-1})$ in (3), we note that the only two possible outcomes are $c_t = c_{t-1} + 1$ when there is no changepoint at time step t, and $c_t = 0$ otherwise. Hence, this is a prior probability on the "lifetime" of this particular segment where the segment ends if a changepoint occurs. In survival analysis, such situations are routinely modeled using a hazard function, which represents the probability of failure in a unit time interval conditional on the fact that failure has not already occurred. If $H(.)$ is a hazard function, the transition probability can be modeled as

$$p(c_t | c_{t-1}) = \begin{cases} H(c_{t-1} + 1) & \text{if } c_t = 0 \\ 1 - H(c_{t-1} + 1) & \text{if } c_t = c_{t-1} + 1 \end{cases} \tag{4}$$

In the special case where the length of a segment is modeled using an exponential distribution with time scale λ, the probability of a changepoint at every step is constant [13], so that $H(t) = 1/\lambda$ and the transition probability is simply

$$p(c_t|c_{t-1}) = \begin{cases} \frac{1}{\lambda} & \text{if } c_t = 0 \\ 1 - \frac{1}{\lambda} & \text{if } c_t = c_{t-1} + 1 \end{cases} \tag{5}$$

The justification for using a uniform prior on changepoints is that apriori we truly cannot have any expectations regarding the time spent by the robot in any specific place. If the robot simply peeks into a place and moves away, we get a very short interval between changepoints while the opposite case where the robot stays in one place for a very long interval is also quite possible. A prior which reduces the chances of a changepoint occurring at a particular time instant risks affecting the result if this expectation is invalid. Instead, by keeping a constant changepoint probability at each timestep and letting the measurements decide the actual changepoint occurrence, we avoid such a risk.

4.2 The Data Likelihood

The data likelihood from (2) can be calculated only if we know the distribution parameter to use. Hence, we integrate over the parameter value using the parameter prior

$$p(y_t|c_t, y_{1:t-1}) = \int_{\xi^c} p(y_t|\xi^c)p(\xi^c|c_t, y^c_{t-1}) \tag{6}$$

where ξ^c is the model parameter for segment c_t, and y^c_{t-1} is the data from the current segment. The above integral can be computed in closed form if the two distributions inside the integral are in the conjugate-exponential family of distributions. Expensive numerical integrations have to be employed otherwise. In the following exposition, we assume the conjugate case.

While we call (6) the likelihood of y_t in keeping with our formulation of the problem, it is in actuality the predictive posterior distribution of the current segment since the second term inside the integral is the posterior distribution on parameters for the current segment. The intuition for our changepoint detection algorithm can be seen from this relationship. If the predictive posterior for the current segment yields a low value for the current measurement, it is likely that a changepoint has occurred. On the contrary, if the prediction for y_t is good, it implies a continuation of the same segment and the same parameter regime.

Now, further, let the integrated function from (6) be denoted as $p(y_t|c_t, \eta^c_t)$ where η^c_t parameterizes the integrated predictive posterior. Even though this function is usually not in the exponential family, it may be possible to update it directly using the sufficient statistics of the data corresponding to the current segment $\{y^c_{t-1}, y_t\}$, i.e. the integration need not be performed at every step. In the case where t is a changepoint, the function is computed with prior values for $\eta^{(0)}$ (since $c_t = 0$) instead of any sufficient statistics.

4.3 Computational Cost

The algorithm described so far is exact. After n measurements, the possible segment lengths range from 0 to n, and the posterior contains the probability of all these cases. Further, if the optimal changepoint locations are also needed, the posteriors from all timesteps have to be kept. Hence, the runtime and memory costs per timestep are $O(n)$ while the total memory cost is $O(n^2)$. These requirements are incompatible with long term online operation. A simple solution is to discard segments with probability less than some small number ϵ from the posterior after each step. This reduces the memory and runtime costs but does not address the worst-case scenario wherein these costs remain the same as before. Hence, we next provide a simple and intuitive particle filtering approximation that exhibits constant runtime and $O(n)$ total memory cost.

4.4 Online operation using particle filtering

We need the locations of changepoints and the exact algorithm above accomplishes this by maintaining the posterior over segment lengths c_t for all t. We now approximate the posterior using N weighted particles, thereby obtaining a constant runtime algorithm.

Combining (2), (3), and (4) we get the segment length posterior as

$$p(c_t|y_{1:t}) \propto \begin{cases} w^{(0)}_t \sum_{c_{t-1}} H(c_{t-1}+1)\rho_{t-1} & \text{if } c_t = 0 \\ w^{(c)}_t \sum_{c_{t-1}} \{1 - H(c_{t-1}+1)\}\rho_{t-1} & \text{if } c_t = c_{t-1}+1 \end{cases} \tag{7}$$

where the particle filter weights are given by

$$w^{(c)}_t = p(y_t|c_t, y^c_{t-1}) \tag{8}$$

and, for the case where t is a changepoint and y^c_{t-1} is the empty set, $w^{(0)}_t = p(y_t|c_t, \eta^{(0)})$, with $\eta^{(0)}$ being the prior parameter value of the integrated likelihood (6). $\rho_{t-1} = p(c_{t-1}|y_{1:t-1})$ is the posterior from the previous timestep.

Clearly, the posterior (7) is amenable to straightforward use in a particle filter with w_t as the particle weights. We use the optimal stratified resampling method of [12] that ensures runtime and memory usage proportional to the number of particles, i.e. $O(1)$ with regard to number of measurements seen so far. The particle weights are given by (6).

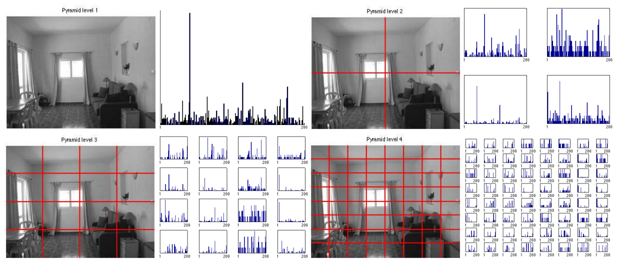


Fig. 4: The Spatial Pyramid histogram: Histograms of clustered SIFT features, computed on image regions at different spatial resolutions, are concatenated to yield the representation of the image. The image regions are obtained by dividing it into successively finer grids.

Note that since the likelihood parameters ξ^c in (6) are integrated out, this particle filter is Rao-Blackwellized [4] and has lower variance than a standard particle filter. This also makes the convergence of the algorithm more efficient.

5 Changepoint detection for video sequences

We model images using a "bag of words" model wherein a histogram is used to represent the image. These histograms are also used as input measurements to the PLISS algorithm. First, in an offline phase, SIFT features are computed on a dense grid on each of a set of images. The features are vector quantized using K-means to create a codebook of pre-specified size. Note that it is not necessary in this step for the image set used to be similar to the test images, though better results are obtained if this is true.

We compute a spatial pyramid [21] from the quantized SIFT features and this is used as input to the changepoint detection algorithm. The spatial pyramid is obtained by computing histograms of feature frequencies at various spatial resolutions across the image. The histogram bins contain the number of features in each of the codebook clusters in the image region under consideration. Following [21], we obtain successive spatial resolutions by diving the image into a grid as shown in Figure 4. Note that only the histograms at the finest resolution need to be computed since the coarser resolution histograms can be obtained by simply adding the appropriate histograms at an immediately finer level. All the histograms from the different grids are then concatenated to yield the spatial pyramid representation. The two parameters for computing the spatial pyramid are thus, the number of levels in the pyramid V and the number of clusters computed in SIFT space (size of the dictionary) K. SIFT features only have local information about an image patch while an image histogram only has global information. By combining both of these at different scales, the spatial pyramid obtains more fine-grained discriminative power.

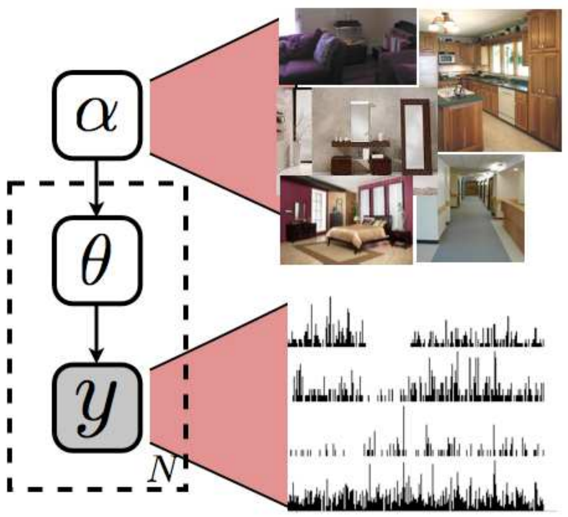


Fig. 5: Graphical model illustrating the Multivariate Polya distribution. To obtain a measurement z, which is a quantized feature histogram, we first sample from a Dirichlet distribution with parameter α to obtain a Multinomial vector θ . This Multinomial distribution is, in turn, sampled to obtain the measurement histogram y. Note that a different θ has to be sampled for each y. Each α roughly corresponds to a place category.

5.1 Multivariate Polya measurement model

A measurement model is required to compute the measurement likelihood in (6). Since the measurements are histograms of word counts, we model them using a multinomial distribution having dimensions equal to the dictionary size. Further the prior over the multinomial parameter is the Dirichlet distribution, which is the conjugate prior and simplifies the computation. Given a histogram measurement y_t, its likelihood according to (6) is

$$P(y_t|\alpha, c_t) = \int_\theta P(y_t|\theta)P(\theta|\alpha, c_t) \tag{9}$$

where $\theta = [\theta_1, \theta_2, \ldots, \theta_W]$ and $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_W]$ are the multinomial parameter and Dirichlet prior respectively, and W is the dictionary size. Assuming that the histogram y_t has bin counts given by $[n_1, n_2, \ldots, n_W]$, the distributions in the integrand above can be written as

$$p(y_t|\theta) = \frac{n!}{n_1!n_2!\ldots n_W!}\theta_1^{n_1}\theta_2^{n_2}\ldots\theta_W^{n_W} \tag{10}$$

$$p(\theta|\alpha, c_t) = \frac{\Gamma(\sum_{w=1}^{W}\alpha_w)}{\sum_{w=1}^{W}\Gamma(\alpha_w)}\theta_1^{\alpha_1-1}\theta_2^{\alpha_2-1}\ldots\theta_W^{\alpha_W-1} \tag{11}$$

The likelihood model in (9), where $P(y_t|\theta)$ is a multinomial distribution (10) and $P(\theta|\alpha, c_t)$ is a Dirichlet distribution (11), is called the Multivariate Polya model [25], or equivalently in document modeling, the Dirichlet Compound Multinomial (DCM) model [22]. The DCM distribution models burstiness in the data, i.e. given that a quantized feature appears once in a document, it is much more likely for it to occur multiple times rather than just once or twice. This is an assumption that intuitively holds for all realistic image data, particularly since we are dealing with densely computed features.

Note that, since we are using spatial pyramids as input, it is only required to model the histograms at the finest level using the Multivariate Polya model since the coarser level histograms are simply summations of these. Thus, for a pyramid with V levels and level $V = 0$ denoting the whole image, the dimensionality of α is just W even though the spatial pyramid histogram itself has dimension $4^V W$.

Performing the integration in (9), we get the final form of the likelihood as

$$P(y_t|\alpha, c_t) \propto \frac{n!}{\prod_{w=1}^{W} n_w} \frac{\Gamma(|\alpha|)}{\Gamma(n+|\alpha|)} \prod_{w=1}^{W} \frac{\Gamma(n_w+\alpha_w)}{\Gamma(\alpha_w)} \quad (12)$$

where $n = \sum_w n_w$, $|\alpha| = \sum_w \alpha_w$, and $\Gamma(.)$ denotes the Gamma function. Graphical intuition for the DCM model is provided by Figure 5. If the likelihood of a set of measurements is to be computed, then n is taken to be the total counts across all measurements, while n_w is the total count for a particular word, or in our case, a specific dense SIFT cluster.

Given the set of D histogram measurements contained in c_t , the maximum likelihood value for α can be learned by using iterative gradient descent optimization. It can be shown that this leads to the following fixed point update for the α parameter[25]

$$\alpha_w^{new} = \alpha_w \frac{\sum_{d=1}^{D} \psi(n_{dw}+\alpha_w) - \psi(\alpha_w)}{\sum_{d=1}^{D} \psi(n_{dw}+|\alpha|) - \psi(|\alpha|)} \quad (13)$$

where $|\alpha| = \sum_w \alpha_w$ as before, and $\psi(.)$ is the Digamma function, the derivative of the Gamma function. Faster, but more complicated, updates using Gauss-Newton iterations also exist [25]. For new segments, we start the iterations with a prior value α_0, which we compute as the maximum likelihood α value of the whole training set.

We use the DCM distribution as the measurement model to compute the likelihood (6), and the α parameter is also updated after each measurement using the iterative rule (13) or the slightly faster Gauss-Newton updates. Thus, we only deal with (12) and do not need to perform the integration of (9). The iterative solution for α can be initialized using the previous value of α at each step so that convergence is usually fast. However, the calculation is still not incremental in nature as it requires all the segment data at each step. Hence, we see a $O(n)$ time complexity for the α parameter update, albeit with a very small constant factor, where n is the length of segment so far. Also, the computation of the digamma function itself is relatively slow.

5.2 A fast Gaussian changepoint algorithm

We overcome the relatively high amount of computation needed for using the Multivariate Polya distribution as the measurement model by projecting the measurement histograms to a low dimensional space and modeling them using a multivariate Gaussian distributions. The theoretical justification for doing this is provided by the seminal result of Diaconis and Freedman [10], who showed that most high dimensional data, when projected down to a uni-dimensional space, follow a Gaussian distribution. This can also observed in practice by noting that many low dimensional data collections follow Gaussian distributions (heights of men, birth weights of babies etc) while high dimensional distributions are rarely Gaussian. This is mainly because Gaussianity demands a high degree of independence between variables (a Gaussian is simply a rotated version of a distribution with independent coordinates) [8]. This independence is easily obtained if the number of variables is very small but becomes progressively harder as the number of variables become larger since it is highly likely that these variable also depend on each other. Diaconis and Freedman only proved their result for uni-dimensional projections though this has been extended to small multi-dimensional spaces recently [27]. While the exact conditions required for the low-dimensional projections to be Gaussian are hard to verify in our case, we find that this assumption works in practice, as we also demonstrate in our experiments.

We project measurements into a low dimensional space by normalizing the spatial pyramid histograms by the total number of features in the image and using Principal Components Analysis (PCA). The PCA projection subspace is learned offline using some training images which can be the same as the ones used to compute the SIFT clusters for computing the spatial pyramid histograms. In practice, we use a 5 to 10 dimensional subspace for projecting down the measurements.

Assuming the projection is to a d-dimensional space, the likelihood (6) can be written using a multivariate normal measurement model as

$$P(y|\Theta) = \int_{\mu,\Sigma} P(y|\mu,\Sigma) P(\mu,\Sigma|\Theta) \quad (14)$$

where Θ are the prior parameters for the multivariate normal distribution, and μ and Σ are the mean and covariance matrix respectively. We use conjugate prior distributions for the unknown mean and covariance parameters. The standard distributions used for this purpose are a multivariate Gaussian on the mean and an inverse-Wishart distribution, a multivariate version of the Gamma distribution, on the covariance

$$p(\Sigma) = \text{Inv-Wishart}(\Sigma|\Lambda_0, \nu)$$
$$p(\mu|\Sigma) = \mathcal{N}\left(\mu; \mu_0, \frac{\Sigma}{\kappa_0}\right)$$

and $\Theta = \{\Lambda_0, \nu, \mu_0, \kappa_0\}$ is the prior parameter set. We have not given the expressions for these distributions since they are quite involved, especially the inverse-Wishart distribution, and can be looked up easily from any standard reference, for instance [16], pg. 85. The interpretations of the prior parameters are intuitive and again can be found in standard references.

Due to the conjugacy, the integration of (6) can be performed analytically to yield the predictive posterior distribution as a multivariate Student-t distribution

$$p(y_t|\Theta, c_t) = t_{\nu_n - d+1}\left(\mu_n, \frac{\Lambda_n(\kappa_n + 1)}{\kappa_n(\nu_n - d + 1)}\right) \tag{15}$$

where

$$\mu_n = \frac{\kappa_0\mu_0 + n\bar{y}}{\kappa_n}$$
$$\kappa_n = \kappa_0 + n$$
$$\nu_n = \nu + n$$
$$\Lambda_n = \Lambda_0 + S + \frac{\kappa_0 n}{\kappa_n}(\bar{y} - \mu_0)(\bar{y} - \mu_0)^T$$

with n being the length of segment thus far $(n = c_t)$, S being the scatter matrix $S = \sum_{i=1}^{n}(y_i - \bar{y})(y_i - \bar{y})^T$, and $\bar{y}$ is the data mean. By maintaining a few statistics, this predictive posterior can easily be updated incrementally so that the changepoint algorithm in this case is much faster than when using the Multivariate Polya distribution. Evaluating the Student-t distribution of (15) can also be performed faster than in the Multivariate Polya case since no digamma evaluations are necessary.

6 Inferring the Place Label

We now turn our attention to the task of computing the place labels given the changepoint posterior which contains the distribution over temporal segments. From (1), the conditional posterior on the place label is $p(x_t^c|c_t, y_{1:t}) = p(x_t^c|y_t^c)$, which we directly model using Gaussian Process Classifiers (GPCs).

We use Gaussian Process classifiers (GPC) as place models. In our prior work [31], we used Multivariate Polya models as place models in addition to using them for changepoint detection. However, these models are more suited to clustering rather than classification and results degrade if the place categories share similarity or overlap due to too much intra-class variation. This is particularly true as the number of classes increases. GPCs, on the other hand, are maximum margin classifiers that are capable of finding decision boundaries even in difficult high-dimensional scenarios such as ours.

The GPC classifies each image in the current segment individually. The label distributions obtained for each of the images are then averaged to obtain the label distribution for the segment

$$p(x_t^c|y_t^c) = \frac{1}{c_t}\sum_{T=t-c_t+1}^{t} p(x_T|y_T) \tag{16}$$

where $p(x_T|y_T)$ is the classification result from the GPC for a single image y_T (the conditioning on the training images has not been shown for simplicity). If we have L place models, the GPC classification result is a discrete probability distribution with L components, and these are averaged as shown above in (16).

While a product of the individual image probabilities would be the theoretically accurate way to obtain the segment label distribution in (16), we find the above to be more robust in practice. This so called "summation hack" has been used in many other areas and some theoretical justification also exists for it's better performance [55].

We use Spatial Pyramid histograms (SPH) computed as explained above in Section 5 as input to the GPC. However, since the dimensionality of the histograms is extremely high, we first project it down to a lower dimensional subspace using PCA, selecting the dimensionality of the subspace so that most of the variance in the training dataset is captured. Empirically, we have found that for a two level histogram using 400 clusters (dimensionality 2000), a 50 dimensional subspace suffices in almost all cases.

6.1 Gaussian process classifiers as place models

A Gaussian Process is a distribution over the space of functions such that the joint distribution over any collection of data points is a Gaussian. GPs can be viewed as probabilistic kernel machines, and hence, can provide not only a mean value prediction but also the uncertainty measured in terms of standard deviation for a test sample. A large standard deviation signals the absence of any training data in the neighborhood of the test sample, and provides an indication of poor generalization.

For performing classification, we assume the availability of N inputs $\mathbf{y} = \{y_{1:N}\}$ and corresponding training outputs $\mathbf{t} = \{t_{1:N}\}$. The covariance function of the GP is then given by the $N \times N$ Gram matrix $K(\mathbf{y}, \mathbf{y})$. Typically, the kernel function has a number of parameters θ, which are also called the hyperparameters of the GP, that have to be learned using the training set. We use the following commonly used kernel function

$$K(y, y') = v_0 \exp\left\{-\frac{1}{2}\sum_{i=1}^{d} w_i \left(y_i - y'_i\right)^2\right\} + v_1 \qquad (17)$$

where d is the dimensionality of the training inputs and $\theta = \{\log v_0, \log v_1, \log w_1, \ldots, \log w_d\}$ are the parameters to be learned from training data.

For performing multi-class classification using GPCs, we closely follow the exposition of Williams and Barber [45] and also use their method in our implementation. Classification is performed by passing the continuous regression output of the Gaussian Process through a soft-max transfer function so that it can be interpreted as the probability of the input belonging to one of m classes. The continuous output of the GP is called the activation at the input value. We use one GP per class label, denoting the input as y_i and activation as x_i^c for class c. The probability of belonging to the cth class, denoted by π_i^c, is thus given by

$$\pi_i^c = \frac{\exp x_i^c}{\sum_{c'} \exp x_i^{c'}} \qquad (18)$$

The Bayesian prediction distribution for an activation value x^* , whose corresponding input is y^*, is given by

$$\begin{aligned} p(x^*|\mathbf{t}) &= \int p(x^*, \mathbf{x}|\mathbf{t}) d\mathbf{x} \\ &= \frac{1}{p(\mathbf{t})} \int p(x^*, \mathbf{x}) p(\mathbf{t}|\mathbf{x}) d\mathbf{x} \end{aligned} \qquad (19)$$

where $\mathbf{x}$ are the activations corresponding to the training data. For a GP, the joint distribution on training data and query point activations is a Gaussian distribution with covariance function given by the Gram matrix on the set $\{\mathbf{y}, y^*\}$. We only consider GPs having mean zero in this discussion as this is sufficient for our purposes. The error model $p(t|x)$ is the multinomial distribution of (10).

It is not possible to perform the integration in (19) analytically. Following [45], we use the Laplace approximation which centers a Gaussian around the maximum of the distribution with the inverse covariance given by the negative Hessian (second derivative matrix) of the integrand $p(x^*, \mathbf{x}|\mathbf{t})$. Since the likelihood is a multinomial distribution, the log-likelihood of the training data is $\mathcal{L} = \sum_{i,c} t_i^c \ln \pi_i^c$, where t_i^c is the probability of the ith training instance taking on class c. Using (18), we get the log-likelihood as

$$\mathcal{L} = \sum_{i,c} t_i^c \left(x_i^c - \ln \sum_{c'} \exp \pi_i^{c'} \right) \qquad (20)$$

Computing the Laplace approximation can be accomplished using a series of Gauss-Newton steps and is notationally complex but straight-forward otherwise. We refer the reader to [45] for the details.

The GP parameters θ are also obtained using a maximum aposteriori method where the posterior to maximize is

$$\begin{aligned} p(\theta|\mathbf{t}) &\propto p(\mathbf{t}|\theta)p(\theta) \\ &= \int p(\mathbf{t}|\mathbf{x}) p(\mathbf{x}|\theta) p(\theta) d\mathbf{x} \end{aligned}$$

The first term of the integrand is the likelihood (20) while $p(\mathbf{x}|\theta)$ is simply the GP prior involving a zero mean Gaussian with the Gram matrix as the covariance. We again use Laplace's approximation on the integrand and perform the integral analytically subsequently. The optimization to obtain the Laplace approximation is even simpler than the previous case and we omit the details again.

The parameters θ are learned during training time and the predictive distribution (19) is evaluated for each input at runtime. Since this evaluation involves a Laplace approximation, it is relatively slow. However, since we do not expect place categorization to be required after every step the robot takes, this system is still suitable for online operation.

For each segment provided by the changepoint detection algorithm, we find the place label by jointly classifying all the measurements in the segment using (19) and (18). This provides more robustness than classifying single measurements as in a single SVM, since in effect, the label probabilities are obtained by votes from all the measurements from the segment so that a few individual misclassifications can be tolerated without affecting the segment label.

7 Unknown place category detection

Detection of an unknown place requires more involved calculation. We previously used statistical hypothesis testing for this purpose [31]. Hypothesis testing is used to show that the data does not arise from any of the models corresponding to known places. At each timestep, L hypothesis tests are performed with each of the L place models to determine if the data arises from a known place. However, hypothesis testing is not scalable with increasing number of place labels and can only

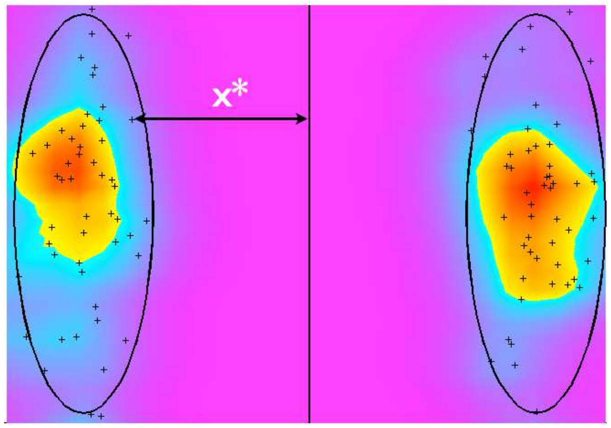


Fig. 6: Hard to categorize inputs (also called perplexing inputs) are close to the margin, i.e. have a small activation value x^* and/or have a large GP uncertainty due to being far away from training data. In the figure, these regions are shown in purple.

be used with probabilistic models such as the Multivariate Polya models used in [31]. As mentioned before, these models do not yield good classification results in difficult environments.

In this work, we take a different approach to detecting unknown places based on the uncertainty provided in the GP output. Firstly, note that the activation of the GP for an input is proportional to the distance to the margin of the classifier, the margin being the classification boundary in kernel methods. Inputs that are close to the margin are harder to classify for the GP while inputs far away from the GP on either side of the margin can be confidently classified. This is similar to a SVM. However, in a GP, we also obtain a large covariance, and hence a low likelihood of the correct classification, when the test point is far away from any training point. Thus, we can say that measurements from unknown place categories will generally either lie close to the margin or far away from any training data. This is illustrated in Figure 6. The covariance of the activation value is obtained through the Laplace approximation, which gives the inverse covariance as the negative Hessian of the integrand in (19).

We detect unknown places using both the activation value and the covariance provided by the GPC. In particular, we use the perplexity statistic

$$p^* = \frac{x^*}{\sqrt{\varSigma^*}} \tag{21}$$

where x^* is the activation value as before and $\varSigma^*$ is the associated covariance value. Note that p^* has small values for difficult inputs. This has previously been used in the context of active learning in computer vision to detect and learn perplexing examples which the classifier is having trouble labeling [18].

We compute the perplexity statistics for misclassified training examples and also calculate their mean and variance. Note that since p^* is small for difficult examples, finding the average over the whole training set would bias the value upwards. We set as threshold the value

$$t^* = p^*_{av} - 2\sigma^*_{av} \tag{22}$$

where p^*_{av} is the average perplexity of the misclassified training examples and σ^*_{av} is their standard deviation. During runtime, any test instance having perplexity value less than t^* is classified as an unknown place category. We only set the segment label to be unknown if a majority of the test instances in the segment have an unknown place category. In such a case, the probability of the unknown place category is set apriori to a distribution for unknown places $p(x|\text{new label})$. This distribution should reflect our knowledge that the segment is more likely to correspond to a new label than to one of the known ones. We do this by setting $p(x|\text{new label})$ such that the probability of the new label is twice the probability of the other known classes. However, various other choices (such as setting $p(x|\text{new label}) = 0.9$) are also possible.

In terms of implementation, we augment the changepoint algorithm described in Section 4 so that the discrete distribution on places is stored with each segment c_t. Similarly, in the particle filter, each particle maintains a place distribution. The place distribution generally becomes increasingly confident as the length of the segment c_t increases and is robust to noisy measurements and outliers. However, since it is also recomputed with each measurement, the algorithm does not make any irrevocable decision with regard to the place label and can "change its mind" given enough evidence. The cost of updating the place distribution is linear in the number of labels and hence, does not affect the runtime of the changepoint algorithm.

A recap of the overall PLISS algorithm using the GP classifier for place models is given in Algorithm 1. This algorithm uses the Dirichlet Compound Multinomial (DCM) model but the changepoint detection part can similarly be computed using the Gaussian ansatz as explained in Section 5.2.

8 Experiments

We present extensive experiments probing the various parts of the PLISS system and comparing it with another state of the art method. Unless noted otherwise, all results were obtained using 50 particles in the particle filter approach. The λ parameter in the transition probability (5) was set to 20 based on cross validation

Algorithm 1 Particle filtering algorithm for PLISS using the DCM (aka Multivariate Polya) model and Gaussian Process classifier

```
1. Initialize: Set prior parameter $\alpha_0$ to be the maximum likelihood value of the DCM parameter on the training set. Learn a GPC on the training set. For all particles, $c_0 = 0$ and $x_0^0 = unif$ (unif. dist. over known labels). Learn perplexity threshold $t^*$.
2. Update particle set:
   For every timestep $t$ do -
   For every particle containing weight, segment length, label distribution, and DCM parameter $\{w_{t-1}, c_{t-1}, x_{t-1}^c, \alpha_{t-1}^c\}$ do
     – Create two new particles (1) No changepoint case $l_1 = \{w_{t-1}, c_{t-1}+1, x_{t-1}^c, \alpha_{t-1}^c\}$ (2) changepoint $l_2 = \{w_{t-1}, 0, unif, \alpha_{t-1}^c\}$
     – Compute the prior for the (4) and update weights of $l_1$ and $l_2$
     – Using $y_t$, learn new parameter $\alpha_t^c$ for $l_1$ using (13) and set parameter $\alpha_t^0$ for $l_2$ to $\alpha_0$
     – Compute incremental weights for $l_1$ and $l_2$ using particle weight definition (8) and DCM likelihood function (12) and multiply with particle weights
3. Resample from weights to get new set of particles
4. Update place distributions:
   For every particle $\{w_t, c_t, x_{t-1}^c, \alpha_t^c\}$ do
     – Compute the perplexity statistic for each test instance in segment and determine if segment is unknown place category
     – If $y_t^c$ is from an existing place category, compute $x_t^c$ using (16)
     – If $y_t^c$ is from an unknown place, create new place label and set $x_t^c$ to the prior distribution $p(x|\text{new label})$
```

performed on a subset of the training data. For using the Gaussian changepoint algorithm, the spatial pyramids were projected down to a 10-dimensional subspace before modeling using a multivariate Gaussian distribution. The initial mean and covariance, which are the priors for the multivariate Gaussian distribution, were computed from all the training data. The prior parameter for the DCM model was also similarly computed. We implemented the system in Matlab with key portions, such as k-means clustering for computing the SIFT visual words, performed in C. All timing results are obtained by running the system on a Macbook Pro laptop with 8 GB RAM.

SIFT features were computed in all cases on a grid having distance 2 pixels between nodes in both x and y directions. Features were computed on 16x16 image patches. The features were clustered to obtain a dictionary of size 1024. We used a three-level pyramid to compute the spatial pyramid histogram in all cases.

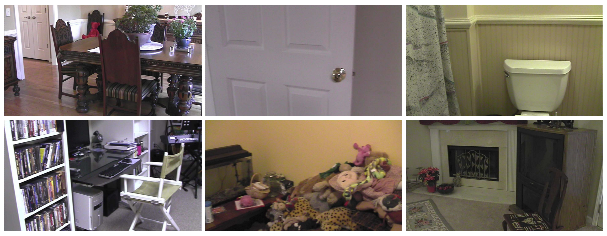


Fig. 7: Some examples of image frames from the VPC dataset. Note the image at top center showing a door and a wall, which is impossible to categorize as anything other than "unknown".

8.1 The VPC Dataset

We use the VPC dataset [46] for almost all our experiments. The dataset consists of image sequences from six different homes, each containing multiple floors. The data is grouped into image sequences from the floors and the image set from each home consists of between 6000 and 10000 frames. In our experiments, we consider sequences from each floor to be a different image sequence. The dataset has been manually labeled into categories to provide ground truth for the place categorization problem. Five categories are present in all the sequences and hence, we only use these labels in our experiments. This also helps facilitate comparison against [46]. In addition, a "transition" category is used to mark segments that do not correspond clearly to any one class. The VPC dataset is especially difficult since no effort has been made to keep all the images in the sequence informative. Thus, a number of images contain only a wall, which is something that could also be expected when a robot is moving around. This is illustrated in Figure 7 through some images taken at random from the dataset.

8.2 Implementation of the VPC algorithm

We implemented the VPC system of [46] which uses CENTRIST features and provides a reference benchmark for the VPC dataset. The system consists of a naive Bayes classifier, the output of which is filtered using Bayesian filtering. We now describe the image representation used by the VPC system.

The underlying features used by the VPC system is CENTRIST, which in turn is based on the census transform [48]. The census transform is a local feature computed densely for every pixel of an image, and encodes the value of a pixel's intensity relative to that of its neighbors. It was originally introduced for identifying correspondence between local patches. The census transform is computed by considering a patch centered at every pixel. The transform value is a positive integer that takes a range of values depending on the size of the patch. For instance, a patch size of 3x3, where there are 8 pixels in the patch apart from the central pixel, yields transform values between 0 and 255 (2^8 values). This is equivalent to having a dictionary size of 256 without the need for the clustering step. Computing a histogram of these 256 values yields the CENTRIST (census transform histogram) descriptor. A big advantage of the census transform is that it is extremely efficient to compute.

The image representation used by the VPC system is obtained by dividing an image into 16 equal segments on a 4x4 grid and computing a CENTRIST descriptor for each segment. The descriptors for each segment obtained on all the images of the training set are then vector quantized using the k-means algorithm, i.e. we obtain 16 CENTRIST codebooks, one for each image segment. The CENTRIST descriptors are then replaced by their corresponding cluster number, so that the final image representation is simply a 16-dimensional vector, each containing a number between 1 and K, where K is the number of clusters in each of the codebooks.

For computing the census tranform itself, we used patch sizes of 3x3 and 5x5. While the natural dictionary size of 256 was used for the 3x3 case, we clustered the census transform obtained from the 5x5 case to obtain 1024 clusters.

8.3 Changepoint detection

We evaluated the changepoint detection by comparing the changepoints detected by the algorithm with the groundtruth in the context of our application, i.e. the time instants when the robot enters or exits a particular place or place category. A subset of the VPC dataset consisting of six image sequences (sequences from six floors of homes) and containing a total of 14,346 frames was used for the evaluation. Changepoints were taken to be video locations where place label changes are marked in the groundtruth labeling of the dataset. The total number of changepoints in these sequences, thus defined, was 76.

Note that, by definition, the algorithm is trying to detect changepoints as locations where the parameters of the model undergo a drastic change whereas the groundtruth defines changepoints as corresponding to changes in place label. Hence, this experiment provides a direct test of the effectiveness of our modeling of the place categorization problem. In this context, it is more important for the algorithm to detect locations where the label changes than to avoid spurious detections. This is because a missed detection (false negative) creates a data segment wherein measurements arising from two types of place labels have been conjoined, which can lead to a large segment of input being misclassified. On the contrary, if the algorithm even wrongly detects a changepoint at each frame in the worst case, it will only default to frame-wise classification. Thus false negatives in the changepoint detection affect the performance of the place categorization more drastically than false positives.

	Correct	False Positive	False Negative	% Correct
SIFT	65	5	11	85.5
CENTRIST (3x3)	56	11	20	73.7
CENTRIST (5x5)	59	5	17	77.6

Fig. 8: Changepoint detection performance for PLISS using various types of features

	Correct	False Positive	False Negative	% Correct
Multivariate Polya model	65	5	11	85.5
Gaussian model	65	6	11	85.5

Fig. 9: Comparison of changepoint detection performance with Multivariate Polya model and it's approximation using the low dimensional multivariate Gaussian distribution.

The changepoint detection performance using various features is shown in Figure 8. SIFT-based image pyramids yield better results than CENTRIST features with underlying census tranform computed on 3x3 and 5x5 patches. All the systems were based on the Multivariate Polya model and a changepoint was declared to have been found if a system fired within 20 frames of the groundtruth, which is about 2 seconds of latency at the experimental frame rate.

Contrary to what we have stated above, the number of false negatives is higher in all cases in Figure 8. This is because a number of the manually marked label changes correspond to short "transition" regions when the robot is moving from one room to another. However, in many instances, our algorithm skips these transition segments and only detects a single changepoint for the motion of the robot from one room to the next. To state this more concisely using an example, the groundtruth is marked as "living room -> transition -> bedroom", thus giving us two changepoints, while our algorithm only detects it as "living room segment -> bedroom segment". This can result in two false negatives and a false positive in the worst case. Finally, we note that many of the changepoints, especially involving "transition" regions, are difficult even for a person to recognize as such and can depend on individual judgement.

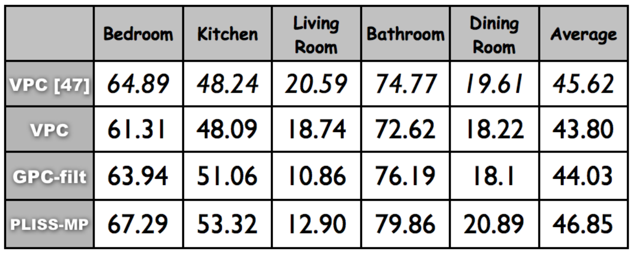
A comparison of the changepoint detection results using the Gaussian model versus the Multivariate Polya model is given in Figure 9. A 10-dimensional subspace was used in the Gaussian case and dense SIFT feature-based spatial pyramids were used in both cases to represent the images. It can be seen that the results are almost identical, proving that our projection into the subspace does not lose any vital information, and nor does the modeling using the Gaussian distribution as opposed to the much more complex high-dimensional Multivariate Polya model. The average runtime per frame was 0.07 and 0.63 seconds using Gaussian and Multivariate Polya distributions respectively. This validates our claim that use of the projected Gaussian distribution for changepoint detection improves efficiency without affecting performance. Note that the runtimes are only for the changepoint detection and do not include place labeling using the Gaussian process classifier.

8.4 Place categorization with and without changepoint detection

We now provide results from our experiments on the complete VPC dataset. We obtained results using the leave one out strategy followed by [46] to facilitate comparison, i.e. the system was trained on labeled data from five houses and tested on the sixth one. The place categorization performance reported here is from the average of having all six houses as the test case in turn. Following [46], only five labels were used for the training - kitchen, bedroom, bathroom, living room, and dining room - and the "transition" label marking frames belonging to none of these place categories was omitted from the classification results.

We first tested the effect of using changepoint detection on the place categorization performance. For this, we implemented a system that performs frame-wise classification using a GPC in the manner described in Section 6.1. The framewise output is then filtered using a Bayesian filtering scheme. We call this system 'GPC-filt'.

Results of these systems are shown in Figure 10 which denotes the PLISS system using the Multivariate Polya model as 'PLISS-MP'. Our implementation of the VPC system was unable to exactly replicate the results of [46], as can be seen in Figure 10, possibly because of engineering issues in the implementation. However, the trends across label types still hold. GPC-filt performs similar to the VPC system while PLISS-MP performs significantly better than our implementation of the VPC system and slightly better than the results reported in [46]. An improvement of 2% is significant since the variance of the averages was less than 0.2% in our experiments. Note that only the frames belonging to the five categories were considered for computing the classification accuracy. The classification numbers for the "Living Room" and "Dining Room" are low because these categories vary the most across image sequences and are easy to confuse with others. This is in part also because there are no distinctive objects in these categories as opposed to, for instance, the "Kitchen" and "Bathroom" classes.



	Bedroom	Kitchen	Living Room	Bathroom	Dining Room	Average
VPC [47]	*64.89*	*48.24*	*20.59*	*74.77*	*19.61*	*45.62*
VPC	61.31	48.09	18.74	72.62	18.22	43.80
GPC-filt	63.94	51.06	10.86	76.19	18.1	44.03
PLISS-MP	67.29	53.32	12.90	79.86	20.89	46.85

Fig. 10: Effect of using changepoint detection for place categorization of known places only. The top row gives the results from [46] while the second row contains results from our implementation of the same system. GPC-filt is the system using frame-wise GPC classification without changepoint detection and PLISS-MP is the PLISS system using the Multivariate Polya model for changepoint detection. All numbers are percentages.

Since the only difference between PLISS-MP and GPC-filt is the changepoint detection (they share the same classifier and use the same features), we can attribute the significant improvement in classification accuracy to the use of changepoint detection. However, the drawback of using changepoint detection is a slower algorithm as compared to Bayesian filtering. Runtime per frame for place categorization was 0.85 seconds for PLISS-MP and 0.27 seconds for GPC-filt. However, this runtime is not too high in absolute terms even when using changepoint detection, and could possibly be made realtime with some optimizations and an implementation completely in C. We believe this is a small price to pay for the consistently higher classification accuracy and increased robustness.

8.5 Unknown place category detection

One of the main advantages of PLISS in relation to most existing methods, including the VPC system, is that it can detect unknown labels. For instance, there are labels, such as "workspace", "family room", and "closet", which are present in only a few sequences of the VPC dataset. The classifier in PLISS and the VPC system do not know about these labels.

We quantified the ability of PLISS to detect unknown places by training the VPC system on an additional category named "transition", in addition to the five classes shown in Figure 10. The "transition" class

	Bedroom	Kitchen	Living Room	Bathroom	Dining Room	Transition	Average
VPC [47]	*64.89*	*48.24*	*20.59*	*74.77*	*19.61*	-	*45.62*
VPC	52.51	41.83	17.88	67.16	14.04	34.37	37.96
PLISS-MP	67.29	53.32	12.90	79.86	20.89	56.62	48.48
PLISS-Gauss	64.61	53.03	12.54	77.59	20.60	53.71	47.01

Fig. 11: Place categorization performance with unknown place category detection. VPC performance (second row) goes down significantly when trained on the transition category that includes all unknown categories. Performance of the PLISS systems is unaffected. All numbers are percentages.

contains all the labels not belonging to any of these five classes. Since this is a catch-all label with wide variation, we do not expect a classifier-based system such as VPC to perform well in labeling this class. The PLISS system does not need to be trained on this "transition" label as it is supposed to detect these as an unknown category. We tested the PLISS system using both the Multivariate Polya model for changepoint detection (PLISS-MP), and the Gaussian changepoint model (PLISS-Gauss).

Training and testing on the VPC dataset were performed similar to the previous section with training done on sequences from five houses and testing performed on the sixth one. Results were averaged across six runs with each of the six houses taken as the test set in turn.

Results of this experiment are shown in Figure 11. Note that results for the VPC system from [46] do not include the transition label since it is not detected therein. Surprisingly, the performance of the VPC system when trained on the "transition" class is much lower than before. The naive bayes classifier underlying the VPC system gets confused by the influx of a new category with a large intra-class variation and it's performance decreases across the board. Indeed, many of the frames marked as "transition" in the dataset actually look into places belonging to the other categories as shown in Figure 13. This causes confusion in the classifier as the classes are no longer as clearly separated as before, so that classification performance is affected.

In contrast, the results of the PLISS system remain unaffected and are, in fact, the same as shown in Figure 10. The PLISS systems were deemed to have recognized a segment as belonging to the transition label when they labeled it as unknown. Thus, the PLISS systems provide significantly better classification accuracy than the VPC system even with the addition of the transition label in the average classification rate. These results provide strong validation for the soundness of our

	Bedroom	Kitchen	Living Room	Bathroom	Dining Room	Transition
Bedroom	67.29	2.11	3.17	6.35	0	21.08
Kitchen	0	53.32	8.27	3.23	1.54	33.64
Living Room	4.52	3.48	12.90	7.31	29.47	42.32
Bathroom	10.84	1.28	0	79.86	0	8.02
Dining Room	3.15	26.76	19.18	1.99	20.89	28.03
Transition	7.26	2.58	15.60	9.79	8.15	56.62

Fig. 12: Confusion matrix corresponding to the PLISS-MP result of Figure 11. All numbers are percentages.

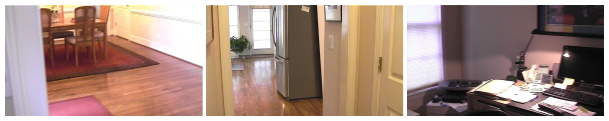


Fig. 13: Images labeled "transition" in the VPC dataset that actually look into places of other categories. This makes learning a classifier for the "transition" label very hard.

strategy for detecting unknown place categories, and also for the usefulness and need for having this ability. We believe that this ability for detecting unknown place categories will be crucial for systems to be deployable in practical scenarios.

A confusion matrix for the PLISS-MP system is shown in Figure 12. Most misclassifications are given a "transition" label. However, the "Living Room" and "Dining Room" categories are often confused with each other, and also with some of the other categories in the case of the latter category. A significant percentage of these misclassifications occur when small video segments, obtained due to false positives in the changepoint detection, are incorrectly classified.

8.6 Choice of image features

We evaluated the use of various low-level image features inside PLISS to determine their effect on accuracy. For this purpose, we implemented three flavors of PLISS based on spatial pyramid histograms using dense SIFT features, and the CENTRIST representation used by VPC with census transform computed on 3x3 and 5x5 pixel image patches. Implementation details of these features are given in Section 8.2 and at the beginning of Section 8.

	Bedroom	Kitchen	Living Room	Bathroom	Dining Room	Transition	Average
SIFT	67.29	53.32	12.90	79.86	20.89	56.62	48.48
CENTRIST (3x3)	61.81	48.27	24.07	65.68	18.18	41.43	43.24
CENTRIST (5x5)	62.28	50.41	23.95	69.16	18.79	44.26	44.81

Fig. 14: Effect of the choice of image features on classification performance. All results (except the VPC system) have been obtained using the PLISS-MP system. All numbers are percentages.

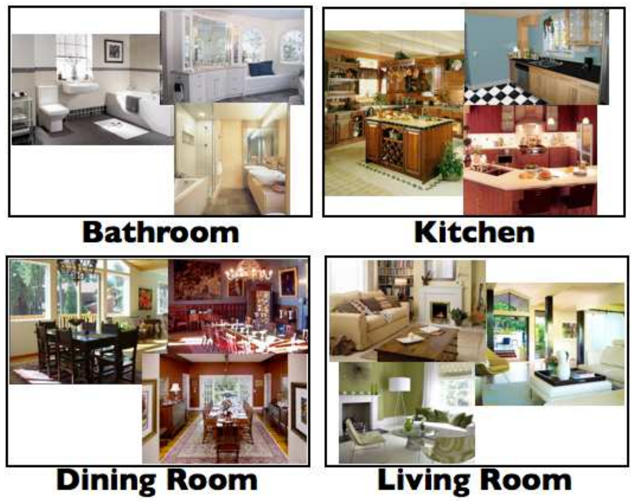


Fig. 15: Examples of images obtained from Google image search used for training place models.

Results on the VPC dataset for the PLISS-MP system are given in Figure 14. We find that dense SIFT performs best and the difference with the Census transform based systems is significant. This is due to the extremely succinct representation used by the VPC system (a 16-dimensional vector for the whole image) which contains much less information compared to the spatial pyramid histogram. However, the results using the CENTRIST representations are still equal to or better than the corresponding VPC results in Figure 11. This demonstrates that while the feature used by us contributes a little to the improved result, the bigger influence is due to the PLISS algorithm.

While [46] also has a comparison of SIFT versus the CENTRIST descriptor, they create 16-dimensional descriptors from quantized SIFT features in an analogous manner to CENTRIST, and conclude that CENTRIST is the better feature. However, our observation here is that the spatial pyramid histogram, which was not used in [46], is a better representation than CENTRIST in the context of PLISS.

	Bedroom	Kitchen	Living Room	Bathroom	Dining Room	Transition	Average
PLISS-MP	41.34	44.28	8.98	44.72	7.04	79.60	37.66
PLISS-Gauss	39.22	39.01	4.83	44.70	6.72	79.58	35.67
VPC	26.92	28.49	8.18	37.41	4.61	-	21.12

Fig. 16: Results from using place models trained on images obtained from Google's image search. Average classification percentage is obtained from six categories for PLISS-Gauss and five categories for SVM-CP. All numbers are percentages.

8.7 Learning from Google Image data

We tested the PLISS system by obtaining training data from the internet. This gives us a sense of how well the system performs when training data is totally disparate from test data. It also ensures that any hidden correlations between image sequences in the VPC dataset do not bias the classification results upwards. We obtained 200 training images for each of the five classes of the VPC dataset by simply typing the label in the Google image search engine and downloading the resulting images. We ensured that irrelevant images are not present in the training set while not removing any relevant image, i.e. no selection bias was present other than to ensure that the image was of the particular place category. We then trained the GPC classifier in the PLISS system on this training set and tested on all of the VPC dataset. A few representative images of each of the categories are shown in Figure 15.

Results for PLISS-MP, PLISS-Gauss and VPC are shown in Figure 16. Classification accuracies here are much less than before due to the extreme difficulty of the problem. However, our aim is to show that PLISS still does something sensible in such scenarios. As expected, large portions of the test data are labeled as unknown place category by PLISS-Gauss, which would be the correct thing to do given the training data. The VPC system was trained only the five relevant categories as obtaining training data for a "transition" category from a Google image search did not make sense and, in any case, would have the made the problem much harder. The VPC results are only on these five categories as is the average accuracy. Even for these categories the accuracy is severely reduced.

9 Discussion

We have presented PLISS, a system for place recognition and labeling based on changepoint detection. PLISS scales well with increasing numbers of place labels, has the significant advantage of being able to detect unknown place categories, is computationally efficient, and has classification accuracy better than state of the art systems. PLISS gets it's robustness from labeling not individual frames, but segments of image sequences, which in turn are obtained probabilistically using changepoint detection. We use the Multivariate Polya model for changepoint detection and have demonstrated that this works effectively on a wide range of image sequences. Our method also encompasses a novel method for detecting the case where the place category of an image or a segment from a video is unknown. This is based on the uncertainty estimate obtained from the Gaussian Process classifier which we use for modeling place categories in a supervised setting.

We have presented extensive experiments on the VPC dataset that yield the following conclusions -

- Changepoint detection using the Multivariate Polya model and the projected multivariate Gaussian model are a suitable means to segment the video sequences into pieces that correspond to specific place categories
- Use of changepoint detection improves the place classification accuracy over frame-wise classification methods that use Bayesian filtering
- Detecting unknown place categories improves the classification accuracy overall, apart from the fact that it is useful, if not essential, in itself.
- PLISS performs better than state of the art place categorization systems in most cases
- Our perplexity measure and the perplexity threshold for detecting unknown place categories, given by (22), perform well in our experiments
- Gaussian Process classifier (GPC) for place modeling, along with our perplexity measure for determining images belonging to unknown categories, is powerful enough to provide reasonable results even on training data that is very disparate from the test data, as shown by our Google image dataset experiment

The basic assumption underlying PLISS is that places are sufficiently distinct to be identified visually. If this is not the case, i.e. the environment is severely perceptually aliased, performance will degrade as with any vision-based system. However, PLISS can be extended to incorporate multiple sensors to overcome such scenarios. We envision significant performance improvements with more sophisticated place models. Such models may even incorporate object and context information from places, which Gaussian Process classifiers cannot do easily.

Using changepoint detection in PLISS, we label large segments that have been identified to be visually homogeneous. In contrast, filtering only takes into account the previous frame's label and so provides an extremely local constraint. In principle, we could extend the Bayesian filtering to take into account the previous n frame labels rather than just one in the following manner

$$p(x_{t-n+1:t}|h_{1:t}) \propto p(h_{t-n+1:t}|x_{t-n+1:t}) \sum_{x_{t'}} p(x_t|x_{t':t-1})p(x_{t':t-1}|h_{1:t-1}) \quad (23)$$

where $t' = t - n$ and h_t is the measurement at time t. This is also a recursive formulation, known as fixed-lag smoothing [34], and is significantly more complicated to implement. Our use of changepoint detection can also be viewed as an adaptive fixed-lag smoothing algorithm wherein n changes with each step. This is clear through a comparison of (23) with equations (2) and (3).

PLISS can learn and update place models online if an online version of the GPC is used [52], a scenario which we have not discussed here. This is helpful in cases where measurements from the same category show variation (albeit only if the variation is gradual). Potentially, PLISS could also learn new categories online if an oracle were available for the first few times that these labels appear. Investigating the online usage of PLISS is future work.

Our future work includes the use of the histogram intersection kernel [17] inside the GPC. Since this kernel forms the natural metric space between spatial pyramid histograms, better results can be expected. However, a problem that needs to be overcome is the high dimensionality of the histograms and the resulting inefficiency of the system. Another area for future work is to incorporate information from multiple sensors, such as depth and range sensors, in addition to images. Use of 3D geometric information and other image features such as edges is also future work. Finally, we would like to evaluate the online learning aspect of PLISS more and improve upon it to produce a system with increased practical use.

References

1. R.P. Adams and D.J.C. MacKay. Bayesian online changepoint detection. Technical report, University of Cambridge, Cambridge, UK, 2007. arXiv:0710.3742v1 [stat.ML].
2. H. Andreasson, A. Treptow, and T. Duckett. Localization for mobile robots using panoramic vision, local features and particle filter. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2005.
3. A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Intl. Conf. on Computer Vision (ICCV)*, pages 1–8, 2007.
4. G. Casella and C.P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, March 1996.

5. A. Taylan Cemgil, W. Zajdel, and B. Krose. A hybrid graphical model for robust feature extraction from video. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.
6. Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.
7. N. Chopin. Dynamic detection of change points in long time series. *Annals of the Institute of Statistical Mathematics*, 59(2):349–366, 2007.
8. S. Dasgupta, D. J. Hsu, and N. Verma. A concentration theorem for projections. In *Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2006.
9. R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (CSUR)*, 40(2), April 2008.
10. P. Diaconis and D. Freedman. Asymptotics of graphical projection pursuit. *Annals of Statistics*, 12:793–815, 1984.
11. S. R. Esterby and A. H. El-Shaarawi. Inference about the point of change in a regression model. *Applied Statistics*, 30(3):277–285, 1981.
12. P. Fearnhead and P. Clifford. Online inference for hidden markov models. *Journal of the Royal Statistical Society: Series B*, 65:887–899, 2003.
13. P. Fearnhead and Z. Liu. On-line inference for multiple changepoint problems. *Journal of the Royal Statistical Society: Series B*, 69(4):589–605, 2007.
14. L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.
15. J. Gaspar, N. Winters, and J. Santos-Victor. Vision-based navigation and environmental representations with an omnidirectional camera. *IEEE Trans. Robot. Automat.*, 16(6):890–898, Dec 2000.
16. A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin. *Bayesian Data Analysis.* Chapman and Hall, 1995.
17. K. Grauman and T. Darrell. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8:725–760, April 2007.
18. A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Gaussian processes for object categorization. *Intl. J. of Computer Vision*, 88:169–188, 2010.
19. B. Kuipers and P. Beeson. Bootstrap learning for place recognition. In *Nat. Conf. on Artificial Intelligence (AAAI)*, pages 174–180, 2002.
20. B.J. Kuipers. The Spatial Semantic Hierarchy. *Artificial Intelligence*, 119:191–233, 2000.
21. S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.
22. R.E. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the dirichlet distribution. In *Intl. Conf. on Machine Learning (ICML)*, pages 545–552, 2005.
23. J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *Intl. J. of Computer Vision*, 43:7–27, June 2001.
24. E. Menegatti, T. Maeda, and H. Ishiguro. Image-based memory for robot navigation using properties of the omnidirectional images. *Journal of Robotics and Autonomous Systems*, 47(4):251–267, 2004.
25. T.P. Minka. Estimating a dirichlet distribution. 2003.
26. O. Martínez Mozos, A. Rottmann, R. Triebel, P. Jensfelt, and W. Burgard. Semantic labeling of places using information extracted from laser and vision sensor data. In *In Proc. of the IEEE/RSJ IROS 2006 Workshop: From Sensors to Human Spatial Concepts*, 2006.
27. A. Naor and D. Romik. Projecting the surface measure of the sphere of l_p^n. *Annales de l'Institut Henri Poincare (B), Probability and Statistics*, 39:241–261, 2003.
28. A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Visual Perception, Progress in Brain Research*, 155, 2006.
29. E. S. Page. Continuous inspection scheme. *Biometrika*, 41:100–115, 1954.
30. A. Pronobis, O. M. Mozos, B. Caputo, and P. Jensfelt. Multi-modal semantic place classification. *Intl. J. of Robotics Research*, 2010. accepted.
31. A. Ranganathan. Pliss: Detecting and labeling places using online change-point detection. In *Proceedings of Robotics: Science and Systems*, 2010.
32. A. Ranganathan and F. Dellaert. Semantic Modeling of Places using Objects. In *Robotics: Science and Systems (RSS)*, Atlanta; USA, 2007.
33. C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, 2006.
34. H. Rauch. Solutions to the linear smoothing problem. *IEEE Transactions on Automatic Control*, 8(4):371–372, 1963.
35. A. Rottmann, O. Martinez Mozos, C. Stachniss, and W. Burgard. Semantic place classification of indoor environments with mobile robots using boosting. In *Nat. Conf. on Artificial Intelligence (AAAI)*, 2005.
36. R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, July 2009.
37. B. Schölkopf, C. J.C. Burges, and A. J. Smola. *Advances in Kernel Methods - Support Vector Learning.* MIT Press, 1999.
38. C. Siagian and L. Itti. Biologically-inspired robotics vision monte-carlo localization in the outdoor environment. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2007.
39. A. Tapus, N. Tomatis, and R. Siegwart. Topological global localization and mapping with fingerprint and uncertainty. In *Proceedings of the International Symposium on Experimental Robotics*, 2004.
40. E.A. Topp, H. Hüttenrauch, H.I. Christensen, and K.S. Eklundh. Bringing together human and robotic environment representations - a pilot study. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Beijing, China, October 2006.
41. A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin. Context-based vision system for place and object recognition. In *Intl. Conf. on Computer Vision (ICCV)*, volume 1, pages 273–280, 2003.
42. G. Tsechpenakis, D. Metaxas, O. Hadjiliadis, and C. Neidle. Robust online change-point detection in video sequences. In *2nd IEEE Workshop on Vision for Human Computer Interaction (V4HCI), in conjunction with the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.
43. I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume 2, pages 1023 – 1029, April 2000.
44. Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.
45. C. K. I. Wiiliams and D. Barber. Bayesian classification with gaussian processes. *IEEE Trans. Pattern Anal. Machine Intell.*, 20(12):1342–1351, 1998.
46. J. Wu, H. Christensen, and J. M. Rehg. Visual place categorization: Problem, dataset, and algorithm. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

47. J. Wu and J. M. Rehg. Where am i: Place instance and category recognition using spatial pact. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008.
48. R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Eur. Conf. on Computer Vision (ECCV)*, volume 2, pages 151–158, 1994.
49. H. Zender, P. Jensfelt, O. M. Mozos, G.-J. Kruijff, and W. Burgard. An integrated robotic system for spatial understanding and situated interaction in indoor environments. In *Nat. Conf. on Artificial Intelligence (AAAI)*, 2007.
50. Y. Zhai and M. Shah. A general framework for temporal video scene segmentation. In *Intl. Conf. on Computer Vision (ICCV)*, volume 2, pages 1111–1116, 2005.
51. Z. Zivkovic, O. Booij, and B. Kröse. From images to rooms. *Journal of Robotics and Autonomous Systems*, 55(5):411–418, 2007.
52. L. Csato and M. Opper. Sparse online gaussian processes. *Neural Computation*, 14(2):641–669, 2002.
53. I. Posner, M. Cummins, and P. Newman. A generative framework for fast urban labeling using spatial and temporal context. *Autonomous Robots*, 26:153–170, 2009.
54. I. Posner, D. Schroeter, and P. Newman. Using scene similarity for place labeling. In *International Symposium of Experimental Robotics*, 2006.
55. T.P. Minka. The 'summation hack' as an outlier model. 2003.