Creating and Using Probabilistic Costmaps from Vehicle Experience

Liz Murphy, Steven Martin and Peter Corke

Abstract—Probabilistic costmaps provide a means of maintaining a representation of the uncertainty in the robot’s model of the environment; in contrast to the ubiquitous assumptive costmaps which abstract this uncertainty away. In this work we show for the first time how probabilistic costmaps can be learned in a self-supervised manner by a robot navigating in an outdoor environment. Traversability estimates garnered from onboard sensing are used in conjunction with colour information from a-priori available overhead imagery to extrapolate the traversability of locations previously traversed by the robot to a much larger area. Gaussian processes are used to predict the traversability at unknown locations in the 2D map, and a number of techniques to deal with heteroscedastic noise and varying confidence in the training data are evaluated. A priori technique to exploit the probabilistic nature of the map in a probabilistic heuristic for A* search demonstrates that planning over these maps can also be done efficiently.

I. INTRODUCTION

An accurate traversability map is a crucial tool for any robot tasked with operating continuously and over a particular area. Traversability is notoriously difficult to characterize a-priori, it is a complex function of both the terrain and the robot’s capabilities. In this paper, sparsely sampled traversability metrics that are estimated by the robot while it navigates an outdoor area are used in conjunction with a-priori available information (overhead imagery), to show that it is possible to generalize from these few data points to a cost map of the entire region. The aim is to make a map that is as true as possible, and in order to achieve this we argue that it is beneficial to maintain a full representation of the uncertainty in the traversability metric and costmap generation processes through to the point of planning paths over the map. As a result, the maps created here are probabilistic costmaps and we use a technique introduced in previous work [1] [2] to show that they are compatible with fast grid-based planning techniques and thus can be used to generate sensible paths quickly.

It follows that the focus on this particular work is on the generation of costmaps. Typically, costmaps used in planning for field robots are deterministic: the operational area is divided into grid cells and each grid cell is assigned a scalar cost proportional to both the assumed terrain in that cell and the assumed cost to the robot of traversing that type of terrain. This standard approach to costmap creation throws away useful information, as both the terrain type and terrain cost estimations are inherently uncertain. However, keeping track of uncertainty is expensive and traditionally viewed as incompatible with fast grid-based planners such as A* and D* which dominate long range path planning — a problem the prior work circumvents.

In this work we use Gaussian Process (GP) regression [3] to model the relationship between the robot’s position and the corresponding pixel colour in an overhead map to the traversability experienced by the robot at a given point. We assume traversability is proportional to the longitudinal slip experienced by the vehicle at a given location. The use of overhead imagery allows us to estimate traversability at locations far removed from the sampled training data, as the correlations between colour in the image and traversability are learned. The use of this model also reduces costmap creation to a single-stage process: the terrain cost and terrain type are now jointly estimated at a given location. GPs are a non-parametric supervised learning technique, they model the relationship between the inputs and outputs of the training data via a latent function with additive Gaussian noise. Basic GP regression uses a homoscedastic noise model, where the noise variance is assumed to be globally constant. This restrictive assumption is often required to make GP regression tractable but is not coherent with the aim of making the probabilistic costmap as accurate as possible; it is unrealistic to assume that estimates that result from sparse and irregularly sampled training data gathered from a moving platform will exhibit globally constant variance. Heteroscedastic regression allows the noise function to be a function of the inputs, but is more challenging than the homoscedastic case in that prediction is no longer analytically tractable, and approximations to the predictive distribution are required. Several approximations [4] [5] [6] [7] exist. A further complication of the GP framework is that all training data is assumed to have equal weight and hence equal influence on the resulting model. The reliability of estimations of traversability resulting from longitudinal slip of the vehicle are highly dependent on the manner in which the robot is driven over the terrain. The method of [8] which allows individual weighting the training data samples so as to dictate their influence on the final result is applied in this work to address slip/traversability reliability concerns.

The primary contribution of this work is that, for the first time, we demonstrate that probabilistic costmaps can be created in a self-supervised manner by a robot operating on real world terrain and subsequently used for planning. The move to ‘real-world’ data facilitated the development of a new costmap creation technique that simplifies that of [9]. This new technique removes the need for terrain classification and relies solely on (faster) GP regression. It is verified that modelling the terrain with heteroscedastic Gaussian processes produces more accurate results than a standard homoscedastic approach, and that confidence estimates of the
II. RELATED WORK

Gaussian Processes have been successfully applied to the estimation of elevation maps in robotic applications [10] [11]. They have also been used to evolve a more accurate representation of structural dependencies in Occupancy Grid mapping [12] and create grid maps at arbitrary resolutions. In an application more akin to the work presented here, [13] relates remote sensing data to the samples obtained in-situ by an autonomous surface rover conducting a comprehensive geological survey of an area, building up an Intelligent Map of surface materials on-the-fly. The most uncertain areas of the map are targeted for exploration. In [14] wheel slip data, colour overhead imagery and altimetry data are combined in a Gaussian Process framework to produce a mobility maps which dictate the maximum speed a given vehicle can safely drive at. Mobility maps are directional, and the superposition of mobility maps can be used for planning. This differs from the approach taken here as it relies heavily on having a priori knowledge of the terrain slope - we model it as a local effect of lesser impact than variation in terrain type; is less concerned with the detection of terrain types, and does not maintain a representation of uncertainty through to the planning stage.

III. TECHNICAL APPROACH

The problem of generating and planning over probabilistic costmaps can be decomposed into 3 sub-problems: estimating traversability, creating the costmap and planning.

A. Estimating Traversability

The traversability of terrain is measured onboard the robot by calculating longitudinal slip \( T_l \). Longitudinal slip is defined as the difference between the wheel velocity \( (\omega R_w) \), \( \omega \) being the angular velocity of the wheels and \( R_w \) the radius of the wheel and \( v \) is the actual vehicle velocity. The slip metric we use is normalized by the command velocity

\[
T_l = \frac{\omega R_w - |v|}{\omega R_w} \tag{1}
\]

resulting in a non-dimensional quantity which is undefined for zero velocity.

The confidence in the slip metric is dependent on the motion of the vehicle

\[
k_l = \left( 1 - \frac{|\omega|}{\max(\omega)} \right) \frac{|v|}{\max(v)} \tag{2}
\]

and is low when velocity is low or angular rotation \( \omega \) is high.

B. Creating the Costmap

GP regression models the relationship between observed data \( D = \{x_i, y_i\} \in \mathbb{R}^D \times \mathbb{R} \) as the sum of an unknown latent function \( f(x) \) and independent noise \( \varepsilon_i \).

\[
y_i = f(x_i) + \varepsilon_i \tag{3}
\]

Under standard homoscedastic regression, it is assumed that \( \varepsilon_i \sim \mathcal{N}(0, \sigma) \), where \( \sigma \) is globally constant. If, however, the noise variance is not constant, \( \varepsilon \) becomes a function of the inputs \( \varepsilon_i \sim \mathcal{N}(0, \sigma(x_i)) \) which leads to heteroscedastic regression.

Placing a GP prior on \( f \) means the predictive distribution \( p(y^*|x_1^*, x_2^*, \ldots x_n^*) \) of the test set \( X^* = \{x_1^*, \\

\ldots, x_n^*\} \) is a multivariate Gaussian \( \mathcal{N}(\mu^*, \Sigma^*) \) with mean

\[
\mu^* = E[y^*] = K^*(K + R)^{-1}y, \tag{4}\]

and covariance

\[
\Sigma^* = \text{var}[y^*] = K^{**} + R^* - K^*(K + R)^{-1}K^{**}T. \tag{5}\]

Here \( K \in \mathbb{R}^{n \times n}, K_{ij} = k(x_i, x_j), K^{**} \in \mathbb{R}^{n \times n}, K^{**} = k(x_i^*, x_j^*), K^* \in \mathbb{R}^{g \times n}, K_{ij} = k(x_i^*, x_j), R = \text{diag}(r) \)

with \( r = (r(x_1), r(x_2), \ldots, r(x_n)) \) and \( R^* = \text{diag}(r^* \) with \( r^* = (r(x_1^*), r(x_2^*), \ldots, r(x_q^*)) \).

The covariance function \( k \) can take any number of forms, but in this work we utilise only the squared exponential function, (although at times in summation with a diagonal noise covariance).

\[
k(x_i, x_j) = \sigma_f \exp(-\frac{1}{2l^2}|x_i - x_j|^2) \tag{6}
\]

Note that \( \sigma_f \) and \( l \) are hyperparameters (\( \theta \)) of the Gaussian process. Their number and nature depends on the choice of covariance function. GP regression requires these hyperparameters to be learned in a training phase, before the GP can be evaluated for test data in a prediction phase. Typically, training involves learning values of the hyperparameters that minimize the negative log marginal likelihood (NLML)

\[
\log p(y|D, \theta) = -\frac{1}{2}y^T(K + R)^{-1}y - \frac{1}{2} \log |K_y| - \frac{n}{2} \log 2\pi \tag{7}
\]

in an optimization process.

In the case of homoscedastic noise the noise function \( r(x_i) \) is \( \sigma \) everywhere, thus Equations 4 and 5 can be evaluated. Heteroscedasticity however, requires knowledge of \( R^* \) — the posterior noise variance at the training points. There are a number of approaches to estimate this quantity, but here we implemented the technique of Kersting [5], where the predictive distribution we seek to estimate is framed as

\[
p(y^*|X^*, D) = \int \int P(t^*|X^*, z, z^*, D). P(z, z^*|X^*, D) \, dz \, dz^*, \tag{8}
\]

the problem term being \( P(z, z^*|X^*, D) \). Given this term, the first term in Equation 8 is Gaussian and can be easily evaluated using Equations 4 and 5. Kersting et al. use a second GP to estimate the second term, and an Expectation-Maximization (EM) loop to estimate a combined Gaussian process wherein the noise levels become ‘observed’ by taking the mean values provided by the second GP.

To be able to apply different weightings to data points we use the technique of Rottmann [8]. Here, two Gaussian processes are again used; the mean of the first GP is used as a latent variable for a second Gaussian process, and is used to bias the GP away from low-weighted training data

\[
P(y^*|X^*, D, \theta) = \int \int P(y^*|X^*, D, \theta, f_{CV}^{GP_2}) \cdot P(f_{CV}^{GP_2}|X^*, D, \theta) df_{CV}^{GP_2}(10)
\]

\[
P_{GP_2}(y^*) \tag{9}
\]
The second GP can be either homoscedastic or heteroscedastic. The standard approach to learning hyperparameters is to use Equation 7 but an alternative technique is via cross-validation [8]. Sundararajan [15] provides a comprehensive review of criteria that can be applied in conjunction with cross validation to learn the hyperparameters. Rottman employs a modified version of the mean-square error criteria (CV) that incorporates individual training data weights \( w_i \) in the estimation of \( GP_1 \) — because the CV error minimizes the deviation from the predictive mean which the first GP seeks to estimate. A weighted version of the negative marginal data likelihood (GPP) is used to learn hyperparameters for \( GP_2 \), where we wish to take the predictive variance into account.

\[
CV(\theta) = \frac{1}{w_n} \sum_{i=1}^{n} w(x_i)(y_i - \mu^*_i)^2 \tag{11}
\]

in the estimation of \( GP_1 \) — because the CV error minimizes the deviation from the predictive mean which the first GP seeks to estimate. A weighted version of the negative marginal data likelihood (GPP)

\[
GPP(\theta) = -\frac{1}{w_n} \sum_{i=1}^{n} \log(p(y_i|x_i, D^i, \theta)) \tag{12}
\]

is used to learn hyperparameters for \( GP_2 \), where we wish to take the predictive variance into account.

We scale the terrain metric confidence \( k_i \) from Equation 2 to a the range \( \{0 \ldots 1\} \) using

\[
w_i = \frac{1}{2} \arctan 2\pi \frac{k_i}{\max(k_i)} - \pi + 1 \tag{13}
\]

All models were implemented in MATLAB using the GPML toolkit [16].

C. Planning on Probabilistic Costmaps

To plan paths over the probabilistic costmaps we employ the Risky Planning technique described in [1]. A set of 36 landmarks are located around the perimeter of the probabilistic costmap and the distance to/from them from every grid cell in the map is precomputed using Dijkstra’s search [17]. Risk-based heuristics allow the user to trade-off gains in search efficiency against the risk of the search returning a suboptimal path. These are used in conjunction with the ALT algorithm [18] to provide a probabilistic heuristic to drive algorithms such as A* or D*. This framework has been shown to produce sensible paths and offer large speedups over the standard A* search with Euclidean distance heuristic operating over a most-likely version of the probabilistic costmap.

IV. EXPERIMENTS AND RESULTS

Experiments were conducted using the rOscar platform [19], a small robotic car shown in Figure 2. To excite the slip dynamics of the vehicle the car was driven along a sinusoidal path at a constant commanded velocity of approximately 5m/s. The slip of the vehicle was calculated at a rate of 1Hz and registered to the map using GPS. Data was collected in two different locations:

- The Everton Park data set spans an area of 287×134 metres. It is a suburban park containing mainly grass and shrubbery but features a concrete basketball half-court. A total of 172 slip measurements were taken over the area and provide the training data for our GP costmap creation process.

- The Graceville data set covers 230×96 metres. It contains 3 distinct areas corresponding to concrete netball courts, short grass of a soccer field and long grass of an untended field. A total of 287 slip measurements were taken over the area.

The two datasets were used to train 5 different GP models:

- **Homoscedastic NLML** A stock-standard GP regressor that uses a squared exponential covariance function with additive Gaussian noise. The hyperparameters of the process are estimated by minimizing the negative log marginal likelihood of Equation 7.

- **Heteroscedastic NLML** The covariance function and hyperparameter estimation are the same as the homoscedastic case, but here we employ the EM method of Kersting to apply heteroscedastic noise to the model.

- **Homoscedastic GPP** Uses a squared exponential covariance function with additive Gaussian noise. The hyperparameters are estimated using cross-validation and the criteria for optimization is Geisser’s Predictive Probability (GPP), as per Equation 12 but with all weights set to 1.

- **Heteroscedastic Unweighted GPP** As per the homoscedastic case, but the EM method is used to estimate heteroscedastic noise.

- **Heteroscedastic Weighted GPP** Rottman’s weighting technique is first applied to learn a GP that biases the mean of a second heteroscedastic GP away from low-confidence traversability measurements, with weighting calculated as per Equation 13 and using Equation 11 to guide hyperparameter learning. The weightings are also used in learning the hyperparameters of the second GP, where GPP is used as the criteria for optimization under cross-validation as in Equation 12.

The input to the GP model \( \{x_i\}_{i=1}^{n} \) is 4-dimensional, corresponding to the locations \( x_i \) and \( y_i \) of vehicle in the map, and two pixel chromaticity values

\[
x'_r = \frac{r^{2.2}}{r^{2.2} + g^{2.2} + b^{2.2}} \quad x'_g = \frac{g^{2.2}}{r^{2.2} + g^{2.2} + b^{2.2}} \tag{14}
\]
where $r$, $g$ and $b$ are the average gamma-encoded (camera output) pixel colour values averaged over a small neighbourhood surrounding location $(x_i, y_i)$ in the overhead image. The output $y_i$ is the value of slip computed onboard the vehicle using Equation 1.

Figure 1 shows the resulting probabilistic costmaps, with the mean and the variance shown over the entire test area for all models on both test sites.

We measure their accuracy in modelling the terrain with two metrics, and used a cross-validation technique on the training data to do this. We performed 20 independent runs, at each iteration separating out 90% of the training data for training the GP, and testing the performance on the remaining 10% using:

- **Mean Squared Error** between the slip measured at the training data points $y_j$ and the predicted slip from the
Table I (a) shows a marked difference in the performance of the models across the two datasets. On the Everton Park dataset, where the raw slip measurements of Figure 1 (a) appear more randomly distributed than the equivalent in the Graceville dataset, the heteroscedastic models perform best. Again, the Unweighted GPP model is best, followed by its weighted equivalent and then the NLML technique. On the Graceville dataset however, the best fit model is the Heteroscedastic NLML model, followed by its Homoscedastic analog. The relatively poor performance of the GPP cross-validation techniques can be explained by examining the training data in Figure 1 (a). The high-slip region corresponding to the smooth netball court has relatively few datapoints on it, it is likely that some batches of data used in the cross-validation contained no points from this region, causing sub-optimal hyperparameters to be learned.

From these results we can conclude that using heteroscedastic GPs produces more accurate traversability maps. Of the 3 heteroscedastic models we considered there is no clear winner, although in the two sites considered we did not derive any added benefit from incorporating a weight proportional to the confidence in our training data into the modelling process. This could be a factor of the confidences not varying enough to influence the model creation process, and is a problem that warrants further investigation. What the results in Figure 1 and Table I confirm, is that GP model selection is an open ended process, and that systematic comparison amongst candidate models is a crucial step in any GP application. Table II shows that for creating costmaps on this scale, heteroscedastic GPs are far more computationally intensive than their homoscedastic counterparts, but not intractable.

Finally, to prove the use of these costmaps in a real world context we ran Risky Planning on the costmap of Figure 1(f). The results are shown in Figure 3, which shows the result, the better the fit.

Table I (b) shows that the Unweighted GPP Cross-Validation technique performs best in terms of minimizing the mean-squared error on both datasets. Performance of the homoscedastic GPP model is extremely poor — this is possibly because choosing hyperparameters using this technique takes into account matching the variance of the predictive distribution as well as minimizing differences in the mean. Looking at Figure 1 (d) we see the mean of the predictive distribution is highly peaked around the location of the data points and the variance is of a much larger scale than the other techniques. The costmaps of the other GPP based techniques in Figure 1 (e) and (f) are heteroscedastic, so the need to match the distribution is a load shared somewhat by the two-stage EM estimation process; the second noise GP should allow a better fit of the distribution at individual training data points purely because the noise variance is allowed to change adding a degree of freedom to the fitting process. Table (b) also shows there is negligible difference in the performance of the homoscedastic and heteroscedastic GPs with hyperparameters learned using NLML. This is not unexpected as this metric only takes into account the mean of the model. The weighted version of GPP cross-validation is outperformed by its unweighted counterpart, although not significantly.

Table I(a) shows a marked difference in the performance of the models across the two datasets. On the Everton Park dataset, where the raw slip measurements of Figure 1 (a) appear more randomly distributed than the equivalent in the Graceville dataset, the heteroscedastic models perform best. Again, the Unweighted GPP model is best, followed by its weighted equivalent and then the NLML technique. On the Graceville dataset however, the best fit model is the Heteroscedastic NLML model, followed by its Homoscedastic analog. The relatively poor performance of the GPP cross-validation techniques can be explained by examining the training data in Figure 1 (a). The high-slip region corresponding to the smooth netball court has relatively few datapoints on it, it is likely that some batches of data used in the cross-validation contained no points from this region, causing sub-optimal hyperparameters to be learned.

From these results we can conclude that using heteroscedastic GPs produces more accurate traversability maps. Of the 3 heteroscedastic models we considered there is no clear winner, although in the two sites considered we did not derive any added benefit from incorporating a weight proportional to the confidence in our training data into the modelling process. This could be a factor of the confidences not varying enough to influence the model creation process, and is a problem that warrants further investigation. What the results in Figure 1 and Table I confirm, is that GP model selection is an open ended process, and that systematic comparison amongst candidate models is a crucial step in any GP application. Table II shows that for creating costmaps on this scale, heteroscedastic GPs are far more computationally intensive than their homoscedastic counterparts, but not intractable.

Finally, to prove the use of these costmaps in a real world context we ran Risky Planning on the costmap of Figure 1(f). The results are shown in Figure 3, which shows the result of planning a path across the map using two different risky heuristics $h = \mu$ (risk of returning a suboptimal path = 50%), and $h = \mu - 2\sigma$ (risk of returning a sub-optimal path = 2.3%). While a thorough evaluation and explanation of the Risky Planning technique can be found in [1] [2], it is a probabilistic technique, the effectiveness of which can only be validated by drawing a number of sample costmaps from the probabilistic costmap and conducting repeated searches to gather average statistics on performance. The result in Figure 3 is the end result of drawing 60 possible costmap
In this work we used traversability data gathered onboard a car-like robot to generate probabilistic costmaps using a number of different Gaussian Process models. The models differed in their approach to handling noise variance across the costmap, and in the manner in which the hyperparameters of the Gaussian Process were learned. The performance of the models was evaluated against two metrics: mean squared error and probabilistic log likelihood. We can conclude that a heteroscedastic approach to terrain modelling worked best on the two test sites we chose to model, but that the method of choosing the hyperparameters proved to be sensitive to inadequacies in the training data. A method of biasing the GP models away from training data points considered unreliable was also trialled, but showed no improvement over the standard case on our 2 data sets. Furthermore, the existing Risky Planning technique was applied to the generated costmaps and results comparable to earlier simulation results were obtained.

VI. ACKNOWLEDGEMENTS

The authors would like to thank Axel Rottman for sharing his Weighted Gaussian Process implementation.

REFERENCES