3D Path Planning for a Rotary Wing UAV using a Gaussian Process Occupancy Map

Seng Keat Gan, Kwangjin Yang and Salah Sukkarieh
Australian Centre for Field Robotics (ACFR)
The University of Sydney, Australia
{s.gan, k.yang, salah}@acfr.usyd.edu.au

Abstract

This paper presents a 3D online path planning algorithm for a 6DOF Rotary Unmanned Aerial Vehicle (RUAV) operating in a cluttered environment using a Gaussian Process (GP) occupancy map. Traditional grid-based occupancy maps suffer from the curse of dimensionality for platforms that operate in a high dimensional configuration space. In addition, the grid resolutions, shapes and orientations limit the flexibility of the path planner. A GP occupancy map however performs inference directly on the collected sensor data, removing the need of maintaining a full discrete map. It performs inference about the probability of collision for any query point in continuous 3D space that potentially maximizes the flexibility of the path planner. We embed the GP map approach into our previous work on smooth path planning and control for an aerial vehicle and present results in a simulated cluttered environment.

1 Introduction

It is common in the path planning community that a discretised map, one that holds the joint sensory information about the environment, has to be constantly maintained such that a path planner can easily plan a feasible path given this map. The most straightfoward approach is to discretize the operational area into predefined uniform grids where each grid holds a scalar value denoting the type of features in the grid. A simple and efficient path planner, such as the A*, Probabilistic Roadmap (PRM) or the Rapidly-Exploring Random Tree (RRT), can then be easily applied in conjunction with this map as an occupancy checking tool to yield a collision-free path. This grid based path planning approach has been widely used and successfully demonstrated for real-time applications that operate on 2D planar domains, especially well-suited for ground robots [LaValle, 2006; Ferguson and Stentz, 2006; Garrido *et al.*, 2008].

Figure 1: Example application of 3D collision-free path planning: Demonstration of our RUAV platform in a weed management mission.

However, the same approach becomes much more expensive to implement with the increased dimensionality in the robot operational space. For instance, aerial platforms performing low altitude missions, such as autonomous weed management (Figure 1), animal herding, pipeline inspection and search and rescue, often require their robots to effeciently operate back and forth in a large and possibly growing 3D environment that are very often cluttered with obstacles, such as trees, buildings, powerlines and so forth. In these situations, it is essential to develope an efficient path planner that operate in this higher dimensional space.

There have been various successful approaches to path planning demonstration in 3D space [Hwangbo *et al.*, 2007; Garrido *et al.*, 2008; Carsten *et al.*, 2006; Hrabar, 2008]. Nevertheless, the authors preserve the use of grid-based methods (or variant) as a representation of the environment. The grid-based approach suffers greatly from the curse of dimensionality; the grid density required to reasonably represent the 3D operational environment to a certain resolution exponentially increases. Furthermore, the flexibility of the path plan-

ner is restricted by the resolution, shape and orientation of the grid. Obstacle shapes in reality are often irregular. Complex grid meshes are required to represent these obstacles to a reasonable accuracy, increasing the complexity of the map. Each of these grid cells usually holds a constant label to represent the feature within the cell. The result of this is the non-obstacle regions around obstacle boundaries are often misclassified as obstacles if they are sharing the same grid cell with the obstacle, limiting the possibility of the path planner to access these regions.

We tackle this problem by replacing the traditional grid-based occupancy map approach with a Gaussian Process (GP) occupancy map for path planning. The GP map operates directly on the collected raw laser beam data, which thus no information is lost in general. The GP map is continuous. It acts like a blackbox that outputs the probability of occupancy given any selected query point in continuous space. This grants it the potential to explore all possible paths in the full space when used in conjunction with a continuous path planner.

The work of GP occupancy maps was first presented by O'Callaghan [O'Callaghan *et al.*, 2009], who use Gaussian Process as a Probabilistic Least Square Classifier for reconstruction of a 2D occupancy map based on real-time collected sparse laser beam information. We adapted their approach by extending it to 3D with modifications to suit our application for online path planning.

This paper is organized as follows: Section 2 presents a brief background of Gaussian Process and how it can be used as a tool to generate a continuous occupancy map. Our previous work on path planning is briefly outlined in Section 3. Section 4 demonstrates our application of a GP map to online 3D path planning for an RUAV operating in a naturally cluttered environment. Finally, we detail the limitation of the GP map and discuss future work.

2 3D Gaussian Process Occupancy Map

Section 2 is seperated into five parts. The fundamentals of Gaussian Processes and the selection of appropriate covariance function are firstly discussed in 2.1. Following in 2.2 is the extension of GP to Probabilistic Least-Square Classification. Section 2.3 introduces the process of training the GP and classifier hyperparameters. Section 2.4 and 2.5 respectively detail the extraction of training data sets for online occupancy inference and the method of setting up a safety boundary around occupied locations for robust path planning in the next section.

2.1 Gaussian Processes Fundamentals

Gaussian Processes are a powerful nonparametric tool for regression, based on locally available datasets. It represents a family of distributions over functions, and inference takes place directly in the function space [Rasmussen and Williams, 2006]. It is particularly useful to infer the most likely function value f_* and its corresponding variance for a given test input $\mathbf{x}_*$ when exact observation on $\mathbf{x}_*$ is not available. Furthermore for the case when there is no information about the underlying function, inference with a GP can still be achieved by simply placing a multivariate Gaussian distribution across the function space.

Consider $\chi = \{(\mathbf{x}_i, y_i) | i = 1, ..., n\}$ as a collection of N datasets, where $\mathbf{x}_i \in \mathbb{R}^D$ represents a D dimensional training input and $y_i \in \mathbb{R}$ the corresponding scalar training output (in our case, y_i is +1 or -1 for occupied and non-occupied respectively). Condition on χ, for any selected testing input $\mathbf{x}_*$, the predictive distribution of the function output becomes

$$f_* | \mathbf{x}_*, \chi \sim N\left(\mu_*, \sigma_*^2\right) \tag{1}$$

where mean

$$\mu_* = k\left(\mathbf{x}_*, \mathbf{x}\right)\left[K\left(\mathbf{x}, \mathbf{x}\right) + \sigma_n^2 I\right]^{-1} \mathbf{y} \tag{2}$$

and variance

$$\sigma_*^2 = k\left(\mathbf{x}_*, \mathbf{x}_*\right) - k\left(\mathbf{x}_*, \mathbf{x}\right)\left[K\left(\mathbf{x}, \mathbf{x}\right) + \sigma_n^2 I\right]^{-1} k\left(\mathbf{x}, \mathbf{x}_*\right) \tag{3}$$

$k\left(\mathbf{x}, \mathbf{x}'\right)$ is the GP covariance or kernel function, which effectively sets the correlation between the output of $\mathbf{x}$ and $\mathbf{x}'$ based on trained kernel hyperparameters.

Selection of the appropriate covariance function highly depends on the application. The most commonly used is the stationary Squared-Exponential(SE) kernel

$$k\left(\mathbf{x}, \mathbf{x}'\right) = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\mathbf{x} - \mathbf{x}'\right)^T \mathbf{M}\left(\mathbf{x} - \mathbf{x}'\right)\right) \tag{4}$$

where ${\sigma_f}^2$ is the signal strength, ${\sigma_n}^2$ the Gaussian noise variance and $\mathbf{M} = diag\left[1/l_1^2, ..., 1/l_D^2\right]$ where l_i is the length scale that sets the kernel smoothness characteristic in different dimensions. There exist other covariance functions that outperform Squared-Exponential in terms of modelling non-continous spatial functions such as the Neural-Network demonstrated in [O'Callaghan *et al.*, 2009]. Nevertheless, Squared-Exponential is by far the most computational efficient kernel that suits well our application.

2.2 Probabilistic Least Square Classification

Our aim here is to estimate the probability of collision for any chosen location in 3D workspace for collision-free path planning. There are various classification algorithms available but as described by [Rasmussen and Williams, 2006] and implemented by [O'Callaghan *et al.*, 2009], one simple yet efficient method is to treat the

classification as a GP regression problem and then further "squash" the GP predictions through a cumulative Gaussian sigmoid function to obtain a valid probability

$$p\left(y_i|\mathbf{x},\mathbf{y}_{-i},\theta\right)=\Phi\left(\frac{y_i\left(\alpha\mu_i+\beta\right)}{\sqrt{1+\alpha^2\sigma_i^2}}\right) \tag{5}$$

where α and β are the classifier hyperparameters and they can be trained using the Leave-One-Out cross validation procedure described in the following section. The full method is called Probabilistic Least-Square Classification (PLSC).

2.3 Training GP and Classifier Hyperparameters

Training the GP hyperparameters is effectly setting the appropriate properties for the covariance function. Typical methods are based on maximization of the log of the marginal likelihood

$$\theta_{opt}=\arg\max\left[\ln p\left(\mathbf{y}|\mathbf{x},\theta\right)\right] \tag{6}$$

where

$$\ln p\left(\mathbf{y}|\mathbf{x},\theta\right)=-\frac{1}{2}\mathbf{y}^T K_y\mathbf{y}-\frac{1}{2}\ln|K_y|-\frac{n}{2}\ln 2\pi \tag{7}$$

and $K_y=K\left(\mathbf{x},\mathbf{x}\right)+\sigma_n^2 I$. The optimal solution of this method produces a model that balances between model complexity and data fitness, which is known as Occam's razor principle.

Having trained the GP hyperparameters, the classifier hyperparameters can then be trained using the cross validation approach, based on maximizing the sum of the log Leave-One-Out (LOO) probability

$$\left\{\alpha,\beta\right\}_{opt}=\arg\max\left[\sum_{i=1}^{N}\ln p\left(y_i|\mathbf{x},\mathbf{y}_{-i},\theta,\alpha,\beta\right)\right] \tag{8}$$

where

$$\ln p\left(y_i|\mathbf{x},\mathbf{y}_{-i},\theta\right)=-\frac{1}{2}\ln\sigma_i^2-\frac{\left(y_i-\mu_i\right)^2}{2\sigma_i^2}-\frac{1}{2}\ln 2\pi \tag{9}$$

and

$$\mu_i|\theta=\mathbf{y}_i-\left[K\left(\mathbf{x},\mathbf{x}\right)^{-1}\mathbf{y}\right]_i/\left[K\left(\mathbf{x},\mathbf{x}\right)^{-1}\right]_{ii} \tag{10}$$

$$\sigma_i^2|\theta=1/\left[K\left(\mathbf{x},\mathbf{x}\right)^{-1}\right]_{ii} \tag{11}$$

are the mean and variance after leaving out the training dataset y_i. Note that these hyperparameters are fixed once they are trained.

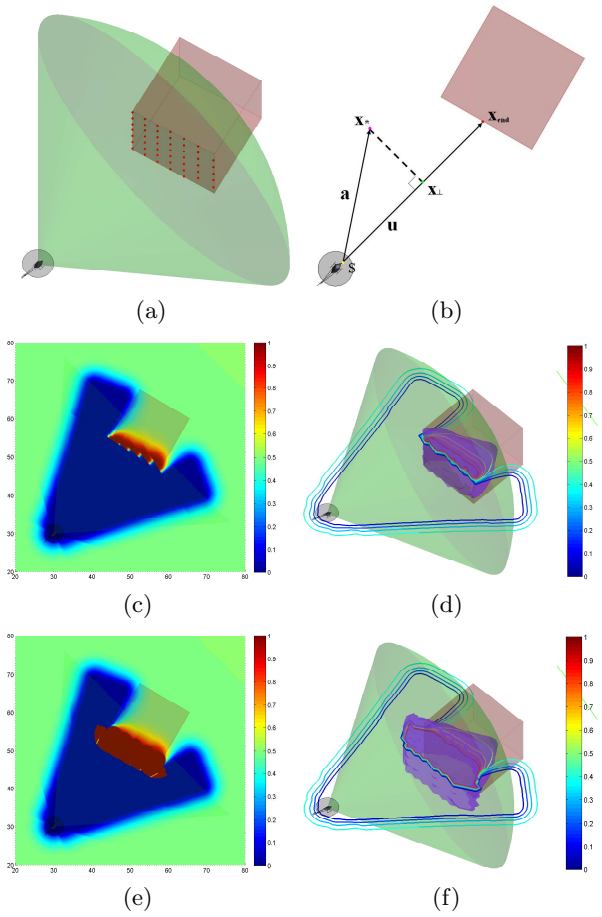


Figure 2: Gaussian Process occupancy map construction. (a) 3D laser sensor. (b) Notation used in Algorithm 1. (c) 2D GP map showing probability of obstacles. (d) 3D GP map obstacle boundary with probability threshold of 0.5. (e) and (f) are similar to (c) and (d) with a 5m safety boundary.

2.4 Sparse Training Data Extraction

It is assumed that the RUAV platform is equiped with a sparse laser range finder, capable of scanning n_b laser beams every instant the sensor is activated, as depicted in Figure 2(a). Each time the sensor scans the environment the following information is stored for each beam segment $\Lambda^k=\left\{\left(\mathbf{s}_i,\mathbf{u}_i,b_i\right)|i=1:n_b\right\}^k$ where $\mathbf{s}_i$, $\mathbf{u}_i$ and b_i are respectively the sensor position, beam vector from sensor to return position and a boolean indicating either 1 if the beam hits an obstacle and 0 otherwise.

Λ is a database of continuous beam segments representing the truth data about the environment. When the occupancy hypothesis of a point $\mathbf{x}_*$ is queried, a series of sparse spatial training data is to be extracted from its local neighbourhood database subset $\Lambda_{\mathbf{x}_*}=\left\{\Lambda_{j,\mathbf{x}_*}|j=1:n_s\right\}$ where n_s is the number of the nearest beam segment. Amongst $\Lambda_{\mathbf{x}_*}$, training input $\mathbf{x}$ and

its corresponding label $\mathbf{y}$ can thus be extracted using Algorithm 1, with notations labeled in Figure 2(b). Line 1 and 2 determine the distance d and location of the point that is projected orthogonally from $\mathbf{x}_*$ onto the beam segment $\mathbf{u}$ (note that $\hat{\mathbf{u}}$ is the unit vector of $\mathbf{u}$). Line 3 extracts the end of the beam segment. Line 4 to 10 simply infer the conditions when the point $\mathbf{x}_\perp$ lies outside of the beam segment. Line 11 and there after is a simple logic that is used to set a safety boundary around obstacles for our path planning purpose that will be further described in the following section. Figure 2(c) and 2(d) depict the 2D and 3D GP map without safety boundary respectively. It can be observed that a sparse laser beam segment is sufficient to represent the sensed obstacle boundary with reasonably well resolution without worrying about the grid resolution, shape and orientation as in the case of a grid-based map.

Algorithm 1 Sparse Training Data Extraction

Require: $\Lambda_{\mathbf{x}_*}, \mathbf{x}_*$
1: $d \leftarrow \mathbf{a} \cdot \hat{\mathbf{u}}$
2: $\mathbf{x}_\perp \leftarrow \mathbf{s} + d\,\hat{\mathbf{u}}$
3: $\mathbf{x}_{end} \leftarrow \mathbf{s} + \mathbf{u}$
4: **if** $d \leq 0$ **then**
5: $\quad \mathbf{x} \leftarrow \mathbf{s}$, $y \leftarrow -1$
6: **else if** $d \geq |\mathbf{u}|$ **then**
7: $\quad \mathbf{x} \leftarrow \mathbf{x}_{end}$, $y \leftarrow 2b - 1$
8: **else**
9: $\quad \mathbf{x} \leftarrow \mathbf{x}_\perp$, $y \leftarrow -1$
10: **end if**
11: **if** $b = 1$ & $|\mathbf{x}_{end} - \mathbf{x}_*| \leq d_{safety}$ **then**
12: $\quad \mathbf{x} \leftarrow \mathbf{x}_*$, $y \leftarrow 1$
13: **end if**

2.5 Safety Boundary Construction

It is common in the path planning literature to include a safety boundary around detected obstacles in order to improve system robustness [Yang *et al.*, 2009]. Real sensory information is always stochastic. The errors in sensor position, direction of laser beam segment and returned range measurement all contribute to obstacle location error. Without considering these errors in detail in our current work, the error of obstacle position could be handled by naively setting a safety boundary around it. This safety boundary requirement is especially crucial when it comes to airborne platforms as flying too close to obstacles significantly increasing the risk of collision. Thus, a safety boundary is necessary and here we propose a simple yet efficient way to include this feature in the GP Map framework, appearing in line 11 to 13 in Algorithm 1.

The trick is to enforce maximum correlation between the output of $\mathbf{x}_{end}$ and $\mathbf{x}_*$ for $\mathbf{x}_{end}$ that is extracted from an obstacle beam when the distance between them is less than the safety boundary threshold d_{safety}. The simplest way to achieve this without modifying the GP framework is to simply set $\mathbf{x}$ the same as $\mathbf{x}_*$, which effectively sets $k(\mathbf{x}, \mathbf{x}')$ to the maximum value of the covariance function, Equation (4).

Traditional planning strategies, such as the grid-based occupancy map framework, are relatively straight forward to include this safety boundary as the full environmental map is continously maintained. The drawback however, is that for every new sensor information, not only the grids that the beam passes through but also the grids within the safety boundary of the obstacle point must be updated, which is an expensive and redundant process and scales badly with an increase in workspace dimension. Our method directly embeds this safety boundary feature in the training dataset extraction process, flexibly controlled by the parameter d_{safety} with no induced cost while transitioning from 2D to 3D workspace. Figure 2(e) and 2(f) demonstrate the changes to the obstacle map after incorporating a 5m safety boundary around detected obstacles.

3 3D Dynamic Path Planning

This section briefly describes our previous work on 3D dynamic path planning and clearly indicates where the GP map fits within it. We firstly justify our choice of RRT as a path planner, followed by path smoothing strategies using Bezier curves taking into account vehicle kinematic and dynamic constraints. We then address replanning strategies based on the probability of collision along the path produced by the GP map, see [Yang *et al.*, 2009] and [Yang and Sukkarieh, 2008] for more detailed information.

3.1 RRT Path Planning

RRT is firstly suggested in [LaValle, 1998] as an alternative to perform path planning. It is a computationally efficient tool for higher degree of freedom configurations. The planning process is a random exploration of the space which means the solution is suboptimal. Having said that, this also means that any point in the full continuous space has the potential of being selected by the planner as part of the path, which is in perfect match to the GP map property for being able to test occupancy condition for any selected location in continuous space.

In Figure 3, the blue branches and red path depict the standard RRT path planning algorithm in a fully known cluttered environment. We apply an efficient path pruning procedure to remove the unnecessary waypoints and form a shorter path than the original RRT path, represented as the green line in Figure 3.

The main requirement in the planning process is the ability to perform collision checks. Given a path, the

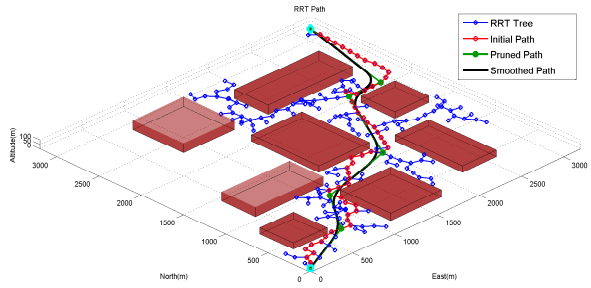


Figure 3: This figure demonstrates the results of RRT path planning and curvature continuous cubic Bezier path smoothing in a fully known cluttered environment.

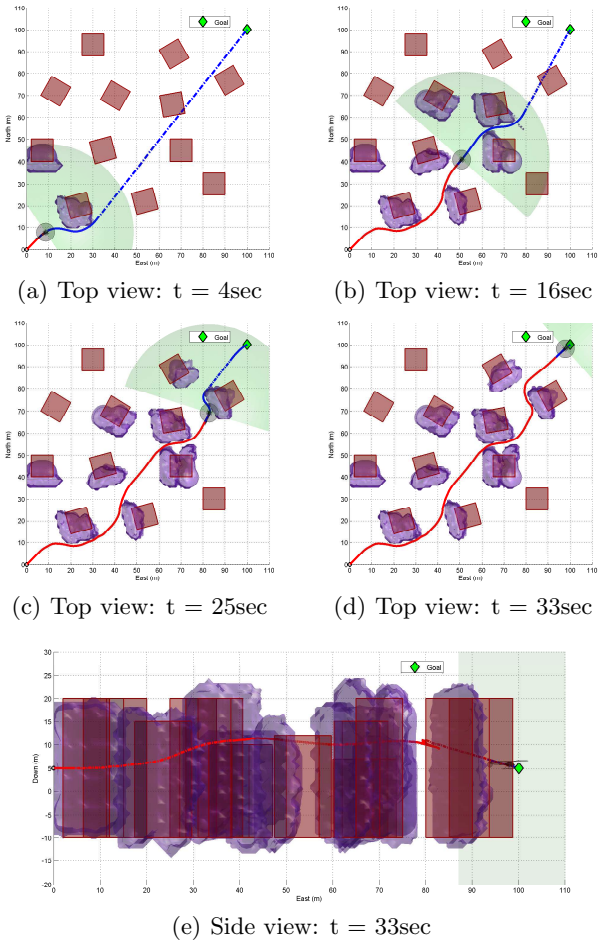
grid-based methods discretise the path into n points, associate these points to their nearest grid cell, and decide if a point is collision-free based on the status in that cell. The process is similar in the GP map approach, but instead of associating points into cells, the query points are directly passed into the GP map collision checker that extract corresponding sparse training data, apply GP regression using Equation (2) to (3), transform into valid probabilities through Equation (5) which are further classified as collision or non-collision based on a user defined threshold.

The entire map is fully described by the collected sensor database, covariance function and the hyperparameters. Its main functionality is to perform collision check. Whenever the path planner requests the occupancy condition of a query point, the local dataset about this point is extracted, followed by GP regression and sigmoid function to produce a valid probability of collision. A threshold is then placed over this probability to obtain a binary class label, i.e. *collision* or *free*.

The pruned path in Figure 3 is piecewise linear and thus is not a suitable path for a RUAV with kinematic and dynamic constraints. To achieve curvature continuity, the position, heading and curvature values of the connecting linear paths must be the same at the joints. A continuous curvature curve which satisfies the maximum curvature constraint can be obtained using the two cubic Bézier spiral curves. With proper design parameter setting [Yang and Sukkarieh, 2008], the generated path is guaranteed to have position, heading and curvature continuities that satisfy the platform maximum curvature constraint. The final smoothed path is illustrated in Figure 3.

4 Simlation Results

In this section we present our GP map based online path planning results for a RUAV in a simulated environment. The RUAV platform dynamics are represented by a 6DOF small helicopter model suggested by Mettler et al. [Mettler *et al.*, 1999]. High performance path



(a) Top view: t = 4sec

(b) Top view: t = 16sec

(c) Top view: t = 25sec

(d) Top view: t = 33sec

(e) Side view: t = 33sec

Figure 4: Above diagrams depict the snapshots taken for a 6DOF RUAV performing online 3D path planning and following in a cluttered environment from the origin to a goal position located at [100,100,5].

tracking is managed using a linear explicit Model Predictive Controller (MPC) previously developed in [Yang *et al.*, 2009].

The environment is sparsely cluttered with obstacles, displayed as the red blocks in Figure 4. The RUAV starts at the origin with no prior information about the environment and the goal is located at [100,100,5]. The RUAV is operating at a constant speed of 5m/s with a fixed 3D laser sensor operating at 1Hz, having a maximum range of 40m and a field of view of 180 degrees in both pan and tilt directions, consisting a total of 121 laser beams uniformly distributed within it.

Figure 4(a) to 4(d) illustrate four snapshots of path replanning after the sensor detects obstacles along the path ahead of the vehicle. In here we use a threshold of 0.6 to further classify the probability of collision into binary labels, i.e. a path can exist in any region that has a

probability value less than 0.6. Note that the replanned paths (dashed blue) in Figure 4(a) and 4(b) penetrate through some obstacles on the field, which is inevitable as these obstacles have not been detected. It is important to choose a threshold that includes low information areas into the planning region during the platform exploration stage. But for some stages where the platform needs to revisit an area that it has been previously explored, a more robust path can be obtained by lowering this threshold value, effectively limiting the planned path to lie within the previously explored regions. Figure 4(e) is to prove that the map and planner are able to operate in the full 3D workspace. Although not shown in here, the planner has the ability to plan a path the goes over the top of an obstacle.

5 Conclusion and Future Work

This paper has explored the use of Gaussian Process occupancy maps as a more promising tool for path planning in 3D space. Results have been demonstrated with application to a RUAV performing 3D online path planning in simulated cluttered environments of different complexity. Our final goal is to embed these algorithms and demonstrate them on our RUAV platform performing real missions.

Ongoing work includes research on robust information management for the GP map. The ability of Gaussian Process performing inferences is based on the collected training datasets. Applying GP map in an online fasion is expected to have a growing database along with the perception of new sensory information. Without proper information management, the GP database will grow without bound which eventually leads to memory failure. A possible research area is to apply information theory to this problem, i.e. only the most informative datasets are to be retained in memory, actively removing redundant information. Furthermore, information rich sensors such as a monocular vision camera could be introduced to further classify the obstacles into dynamic and static classes, which then the covariance function can be modified to include additional temporal correlations between datasets from different timeframe to account for over confidence while performing GP inference in a dynamic obstacle environment.

References

[Carsten *et al.*, 2006] J. Carsten, D. Ferguson, and A. Stentz. 3D Field D: Improved Path Planning and Replanning in Three Dimensions. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3381–3386, 2006.

[Ferguson and Stentz, 2006] D. Ferguson and A. Stentz. Using interpolation to improve path planning: The Field D* algorithm. *Journal of Field Robotics*, 23(2):79–101, 2006.

[Garrido *et al.*, 2008] S. Garrido, L. Moreno, and D. Blanco. Exploration of a cluttered environment using Voronoi Transform and Fast Marching. *Robotics and Autonomous Systems*, 56(12):1069–1081, 2008.

[Hrabar, 2008] S. Hrabar. 3D path planning and stereo-based obstacle avoidance for rotorcraft UAVs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*, pages 807–814, 2008.

[Hwangbo *et al.*, 2007] M. Hwangbo, J. Kuffner, and T. Kanade. Efficient two-phase 3d motion planning for small fixed-wing uavs. In *2007 IEEE International Conference on Robotics and Automation*, pages 1035–1041, 2007.

[LaValle, 1998] S.M. LaValle. Rapidly-exploring random trees: A new tool for path planning. *In*, 1998.

[LaValle, 2006] S.M. LaValle. *Planning algorithms*. Cambridge Univ Pr, 2006.

[Mettler *et al.*, 1999] B. Mettler, M.B. Tischler, and T. Kanade. System identification of small-size unmanned helicopter dynamics. In *Annual Forum Proceedings- American Helicopter Society*, volume 2, pages 1706–1717, 1999.

[O'Callaghan *et al.*, 2009] S. O'Callaghan, F. T. Ramos, and H. Durrant-Whyte. Contextual Occupancy Maps using Gaussian Processes. In *2009 IEEE International Conference on Robotics and Automation,(ICRA 2009). Proceedings*, 2009.

[Rasmussen and Williams, 2006] C.E. Rasmussen and C.K.I. Williams. Gaussian processes for machine learning. 2006.

[Yang and Sukkarieh, 2008] K. Yang and S. Sukkarieh. 3D smooth path planning for a UAV in cluttered natural environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*, pages 794–800, 2008.

[Yang *et al.*, 2009] K. Yang, S.K. Gan, and S. Sukkarieh. An Efficient Path Planning and Control Algorithm for RUAV's in Unknown and Cluttered Environments. In *International Symposium on Unmanned Aerial Vehicles, 2009. UAV09*, 2009.