Receding Horizon Control of UAVs using Gradual Dense-Sparse Discretizations

Petter Ögren* and John W.C. Robinson†

Swedish Defence Research Institute (FOI), SE-164 90 Stockholm, Sweden

In this paper we propose a way of increasing the efficiency of some direct Receding Horizon Control (RHC) schemes. The basic idea is to adapt the allocation of computational resources to how the iterative plans are used. By using Gradual Dense-Sparse discretizations (GDS), we make sure that the plans are detailed where they need to be, i.e., in the very near future, and less detailed further ahead. The gradual transition in discretization density reflects increased uncertainty and reduced need for detail near the end of the planning horizon.

The proposed extension is natural, since the standard RHC approach already contains a computational asymmetry in terms of the coarse *cost-to-go* computations and the more detailed *short horizon plans*. Using GDS discretizations, we bring this asymmetry one step further, and let the short horizon plans themselves be detailed in the near term and more coarse in the long term.

The rationale for different levels of detail is as follows. 1) Near future plans need to be implemented soon, while far future plans can be refined or revised later. 2) More accurate sensor information is available about the system and its surroundings in the near future, and detailed planning is only rational in low uncertainty situations. 3) It has been shown that reducing the node density in the later parts of fixed horizon optimal control problems gives a very small reduction in the solution quality of the first part of the trajectory.

The reduced level of detail in the later parts of a plan can increase the efficiency of the RHC in two ways. If the discretization is made sparse by removing nodes, fewer computations are necessary, and if the discretization is made sparse by spreading the last nodes over a longer time-horizon, the performance will be improved.

I. Introduction

Most modern control approaches, for unmanned air vehicles (UAVs) as well as unmanned ground vehicles (UGVs), include a combination of long term and iterative short term planning.[1–9] In these approaches, the objective of the long term planning is to choose a reasonably short path and avoid running into dead ends, while the objective of the iterative short term planning is to provide locally efficient, safe and collision free trajectories. Thus, the long term planner needs information on the connectivity of the state space in terms of available airspace, obstacle free space or road networks, and the short term planner needs detailed knowledge of the immediate surroundings of the vehicle, as well as a motion model of the vehicle itself. To deal with uncertainties and changes in the environment, a feedback loop is created by iteratively solving the short term planning problem. This is done in such a way that a new plan is created while the first fraction, or phase, of the previous plan is executed, as illustrated in Figure 1. We will refer to these approaches as Receding Horizon Control (RHC).

In many RHC approaches, an Optimal Control Problem (OCP) formulation is used to perform the iterative short term planning.[6–11] Such approaches are quite natural, since many unmanned vehicle problems can be posed as minimum time problems with safety constraints. In Kuwata et al.,[6] a RHC-OCP approach is used to generate low flying trajectories, where a visibility graph is used for the long term plan while the short term plan is produced using Mixed Integer Linear Programming (MILP). An experimental validation

*Senior Scientist, Department of Aeronautics and Systems Technology, petter.ogren@foi.se

†Deputy Research Director, Department of Aeronautics and Systems Technology, Member AIAA, john.robinson@foi.se

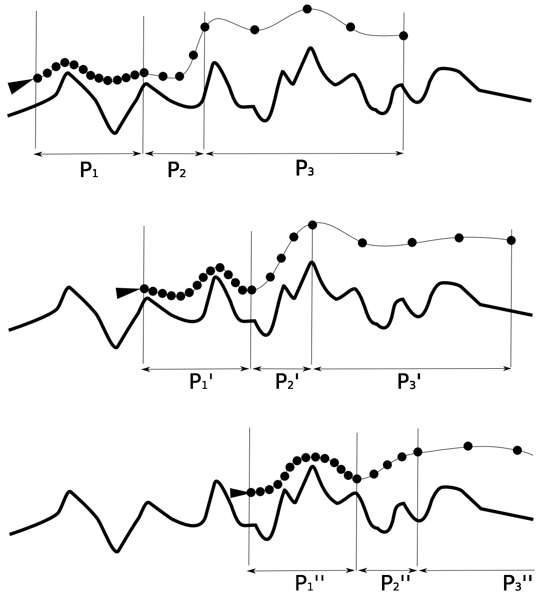

Figure 1. Three consecutive snapshots of a low flying aircraft over mountainous terrain. Each plan consists of three phases P_1, P_2, P_3, ranging from P_1 with a dense discretization and small obstacle margins to P_3 with sparse discretization and large obstacle margins. While P_1 is executed, a new plan P_1', P_2', P_3' is computed and thus P_1 is followed by P_1' instead of the old and sparse P_2. The benefits of P_2 and P_3 are illustrated in Figure 2 below.

of the MILP/RHC framework can be found in Richards et al.,[12] while a similar approach, extended to also account for more elaborate terrain, including threat areas, can be found in Mettler et al.[7] The problem of low flying UAVs was also studied in Anisi et al.,[8] where the long term planning was done on a multi layer grid, and the short term planning was performed using a general nonlinear optimization approach.

The RHC-OCP formulations are however not used in many of the most successful contenders of the Darpa Urban Challenge.[1–5] Instead, they use RHC approaches where the underlying OCP is implicitly addressed using very coarse and robust discrete algorithms, such as Rapidly Exploring Random Trees (RRT),[5] *Goal selection*[4] and *Hybrid A**.[3] As robustness is a key feature in the Darpa Urban Challenge, the choice of not using explicit OCP solvers is presumably due to the lack of speed and robustness in many OCP formulations.

Recently, many ideas towards improving the OCP solvers have been suggested. The issue of real time solutions of OCP using direct pseudo-spectral methods has been investigated,[13,14] as well as the theoretical foundation of such methods and their connection to the continuous problem.[15,16] It is well known that the size and shape of the grid has a major impact on computational efficiency of a numeric solver[17–19] and the issue of grid selection in RHC-OCP formulations has also been studied.[20] In particular, Binder et al.[21] proposed a general method for adaptively refining the grid in moving horizon optimization problems using Galerkin discretizations.

The concept of Sparse-Dense discretizations was introduced by Kumar and Seywald.[22] They showed that a computationally challenging flight path planning problem could be iteratively solved with a speed that allows online implementation and a negligible optimality gap. The dense-sparse approach was built upon by Jain et al.,[19] where an optimal target intercept problem was studied. As in Kumar and Seywald, the approach was iterative, but not RHC, since the OCP was formulated over the whole time interval. Throughout this paper, we denote by Gradual Dense-Sparse (GDS) a discretization that contains more than two levels of density, as depicted in Figure 4.

In this paper, we argue that GDS discretizations are a very natural part of an RHC framework. As noted above, the RHC approaches use plans that are detailed in the near future and coarse in the far future for the following reasons: 1) *It is only the near future parts of the plans that has to be immediately executed. Far future plans can be revised or refined later.* 2) *Information about the vehicle and its surroundings is less uncertain in the near future.* Both arguments apply not only to the division between OCP solution and cost-to-go computations, but to the OCP itself. Using a GDS discretization, we bring the computational asymmetry underlying RHC into the OCP as well, as illustrated in Figure 1. Furthermore, an additional argument for bringing GDS discretizations into the the RHC is the observation made by Kumar and Sey-

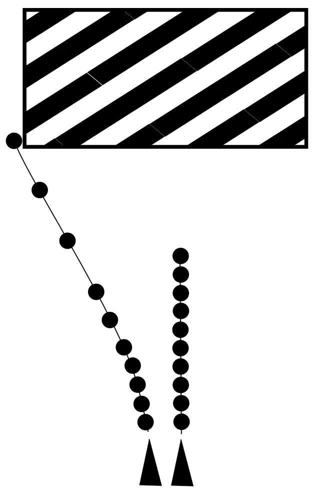

Figure 2. The leftmost aircraft is able to adapt to the obstacle and turn earlier, due to a longer look ahead distance afforded by the sparsity of phase P_2 and P_3 in a GDS discretization. A simulation corresponding to this illustration can be found in Figure 5.

wald:[22] 3) *A reduction in node density in the later parts of fixed horizon optimal control problems gives a very small reduction in the solution quality of the first part of the trajectory.* This fact can be exploited in two ways in an RHC setting: either by removing nodes and thus reduce computation times, or by increasing the horizon length, and thereby improve system performance. While the benefits of less computations is obvious, the implications of an increased planing horizon length is illustrated in Figure 2 and investigated in detail in Section IV.

The outline of the paper is as follows. The trajectory optimization problem is presented in Section II and in Section III the proposed solution is described in detail. In Section IV some simulation examples are given and conclusions are offered in Section V.

II. Problem Formulation

As noted above, many unmanned vehicle control tasks can be formulated in terms of an optimal control problem (OCP), such as the one below.

Problem 1 (OCP) *Solve the following problem*

$$\begin{array}{rrcl} \min_u & J_G & = & \int_0^{T_G} L(x,u)dt + \Psi_G(x(T_G)) \\ s.t. & \dot{x} & = & f(x,u) \\ & g(x,u) & \leq & 0 \\ & x(0) & = & x_{0G} \\ & x(T_G) & \in & S_G \end{array}$$

where u is the control, x is the vehicle state, T_G is the free global final time, $L(x,u)$ is the trajectory cost, $\Psi_G(x(T_G))$ is the final state cost, $f(\cdot,\cdot)$ represents the vehicle dynamics, $g(\cdot,\cdot)$ the trajectory constraints such as collision avoidance, x_{0G} the stating point and S_G the set of feasible final states.

However, the problem above is known to be NP-hard,[23] when used to model a vehicle moving in three dimensions among a set of polyhedral obstacles. Therefore, the OCP formulation is not very suitable for uncertain and dynamic environments requiring rapid computations. A standard way of addressing this issue is to use an RHC approach[7] i.e., first compute a scalar cost-to-go function $\Psi_{CTG}(x)$ and then solve an optimal control problem similar to the OCP above, but over much shorter horizon T and with the new cost-to-go as terminal cost. In fact, if $\Psi_{CTG}(x) = J^*$, the optimal solution cost of the OCP, then the solutions to the OCP and the RHC-OCP below will overlap.[24] This problem formulation can be summarized as follows.

Problem 2 (RHC-OCP) *Solve the problem*

$$\begin{array}{rrcl} \min_u & J & = & \int_{t_0}^{t_0+T} L(x,u)dt + \Psi_{CTG}(x(t_0+T)) \\ s.t. & \dot{x} & = & f(x,u) \\ & g(x,u) & \leq & 0 \\ & x(t_0) & = & x_0 \\ & x(t_0+T) & \in & S \end{array}$$

where most of the notation is borrowed from Problem 1 above, but t_0 is the start time of the current plan, T is the horizon length, Ψ_{CTG} is the cost-to-go, x_0 is the current vehicle state and S includes any desired constraints on the final state of the short term plan regarding e.g., safety.[8]

The general unmanned vehicle RHC algorithm can then be written as follows.

Algorithm 1 (RHC)

1. *Solve the RHC-OCP.*
2. *Apply the control $u(\cdot)$ from time t_0 to $t_0 + T_1$, where $T_1 \leq T$.*
3. *If $x(t_0 + T_1) \in S_G$ then exit.*
4. *Else, set $t_0 \leftarrow t_0 + T_1$ and $x_0 \leftarrow x(t_0 + T_1)$.*
5. *Goto 1.*

Note that ideally, the part of the plan that is beyond T_1 is never executed, instead, it guides the first part of the plan as illustrated in Figure 2, provides an initial guess for the first part of the plan in the next iteration, and serves as a safety fallback if the next iteration somehow fails. We shall now show how the discretization in Step 1 above can be tailored to the requirements given by the fact that we are indeed inside the iterative RHC-loop 1 - 5.

III. Proposed Solution

In this section, we argue that a good way of solving Problem 2 is to do direct numerical optimization using a GDS discretization. We will first discuss the benefits of GDS discretizations in detail, then describe one way of creating initial guesses in the GDS-RHC loop, and finally remark on different ways of implementing the proposed approach.

III.A. Potential benefits of GDS discretizations

As outlined above, the three main reasons for using GDS discretizations are as follows. 1) It is only the near future parts of the plans that has to be immediately executed. Far future plans can be revised or refined later. 2) Information about the vehicle and its surroundings is less uncertain in the near future. 3) A reduction in node density in the later parts of fixed horizon optimal control problems gives a very small reduction in the solution quality of the first part of the trajectory. We will now investigate each of these three reasons in detail.

The fact that only the first part of the plans are executed implies the following. If the overall *executed* trajectory is to be near optimal, then the first part must be computed with a high level of detail in order to capture e.g., aggressive maneuvering. Furthermore, a certain node density might be needed to guarantee collision avoidance or some other feasibility issue with respect to modeling errors, see Figure 3. Finally, in cases where the vehicle dynamics is very complex, the vehicle autopilot executing the trajectory in Step 2 of Algorithm 1 might require a very densely calculated trajectory as input. In those cases, the sparse part of the plan can still increase performance, even though it is not dense enough to be used as safety fallback in the way described below.

All uncertainties regarding the motion of the vehicle, as well as measurements and predictions of the position and motion of other objects and vehicles, will in general grow with time. This fact, together with the fact the value of detailed computations based on uncertain data is often questionable, makes the GDS

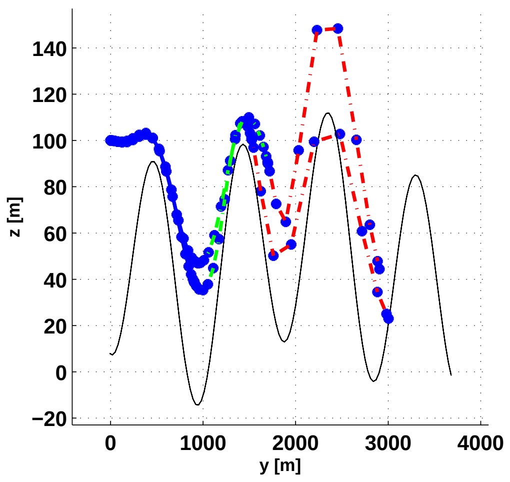

Figure 3. Two examples of a single RHC iteration. The GDS discretization can be seen from the dots denoting collocation points. Both trajectories have three phases, phase 1 (solid blue) that is to be executed, phase 2 (dash-dotted green) and phase 3 (dash-dotted red). As can be seen, an increased obstacle margin is needed to guarantee a collision free trajectory where the collocation points are sparse.

discretization a reasonable choice. An application of GDS-RHC leads to a gradual refinement of the detail in the trajectory from right to left, in any given plan. If the constraints change between planning instances, GDS provides a way to propagate information about the changes backwards from coarser scales at the far right end to finer scales near the left end of the planning horizon. When an old plan is used as a starting guess for the new, GDS therefore provides an economical way of handling the changes since computational power is not wasted on details near the far right which have an uncertain future evolution. At the same time, changes that occur in the midterm future, and can expected to be more certain in the sense that changes are likely to be smaller, are taken more into account.

The third reason, regarding the consequences of a sparse far future plan, is due to an observation made by Kumar and Seywald.[22] They solved a minimum time-to-climb problem for an F-15 aircraft, formulated as a fixed endpoint optimal control problem, in an iterative fashion with an optimality gap of less than 0.5%. The algorithm ran for 46 iterations, each having only eight nodes dispersed in a dense-sparse way from start to goal. After each iteration, the first fraction of the control was applied and a new starting point was found closer to the end point. Thus, the result can be interpreted in terms of how large an impact the discretization density of the last part of the trajectory has on the quality of the first part of that trajectory.

All three arguments above were made to show that a GDS discretization will not *reduce* the performance of the RHC. The *advantages*, on the other hand, are as follows. If the GDS is created by just removing node points, the computations in the RHC will converge much faster.[17,18] If, on the other hand, the far future of the plan is made sparse by extending its endpoint even further into the future, the performance of the RHC will be improved, as illustrated in Figure 2 and shown in Section IV.

The GDS can also be used in schemes that focus on vehicle safety. To guarantee safety, even in the case when the planning process for some reason breaks down, some RHC schemes demand that the trajectory ends in a so-called *safe* state. This can be either a stand still, for ground vehicles or helicopters,[5,9] or a guaranteed obstacle free circular loitering pattern, for fixed wing aerial vehicles.[10,11] During regular operations, these safety-maneuvers are never carried out, as they are not located in the first phase of the trajectories. Thus they can be planned with sparse collocation points and large safety margins. However, to prevent them from hampering performance, the horizon of the plan must be long enough so that their impact on the first phase is limited.

Before ending this section, we note that an extreme version of GDS would be to extend the approach to include not only the discretization, but also the modeling detail. In such an approach, a kinematic model might be used for the far future plan while a dynamic model is used for the near future.

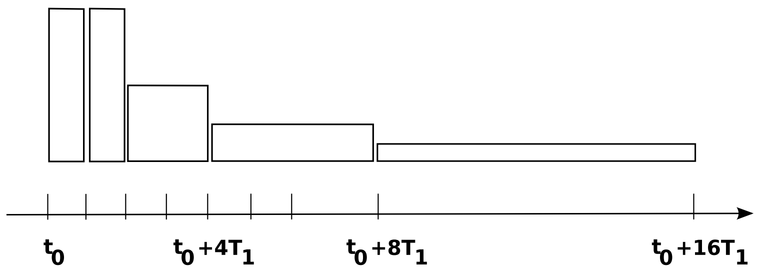

Figure 4. Illustration of the GDS discretization used in the simulations below. This GDS is so-called dyadic, with 4 different discretization densities. Each box symbolizes a phase of the OCP and the height of the box quantifies the density on a linear scale. Thus, if all phases contain the same number of nodes, the length is inversely proportional to the density.

III.B. Iterative refinements to get good initial guesses

When traversing through the steps in Algorithm 1 repeatedly, a new RHC solution is computed over the current planning interval $[t_0, t_0 + T]$ in each pass, and data from the old plan can be reused as initial guess for the new plan. In the setting depicted in Figure 1, data from phase P_2 can be used to produce an initial guess for P_1', data from P_3 can be used to produce an initial guess for P_2' and parts of P_3', and so on.

Creating the new initial guesses can be done in many ways, and here we outline one option, based on multiresolution techniques. The details can be found in the Appendix, and include using the multiwavelet framework of Alpert et al.[25,26] for reconstruction/decomposition in locally polynomial bases, together with a certain dyadic divisioning of the phases and a cyclic strategy for handling the passes in Algorithm 1. The dyadic divisioning scheme is illustrated in Figure 4. We emphasize that the divisioning scheme and accompanying multiresolution technique are not essential for the GDS approach, they merely represent a convenient choice for implementing and updating the discretization which well adapted to the Legendre-Gauss node distributions used in a certain pseudo-spectral OCP solver (see below).

III.C. Implementation of the GDS approach

The GDS approach described in this paper can be used in any implementation of the RHC algorithm where the discretization density can be controlled. We have chosen to use the software package Gauss Pseudospectral Optimization Software (GPOPS).[27] In GPOPS, a problem can be composed of a number of *phases*, where the number of nodes, and extent in time of each phase, can be set by the user. Below we will see how GPOPS is used to solve an example problem involving a low flying UAV.

IV. Simulations

To illustrate the approach we use the following example problem. A low flying UAV is modeled by generalized Dubins car[28] in 3D, which is equivalent to a second order point mass model with constant velocity magnitude and bounded turning rate in both pitch and yaw.

Writing down the corresponding RHC-OCP, $f(x, u)$ is given by

$$\dot{p} = v \tag{1}$$

$$\dot{v} = u, \tag{2}$$

where $x = (p, v) \in \mathbb{R}^6$ is the state and u is the control (normalized control force). The state and control constraint $g(x, u)$ includes

$$u^T v = 0 \tag{3}$$

$$||u|| \leq a_{\max} \tag{4}$$

$$h(p_1, p_2) + h_0 \leq p_3, \tag{5}$$

where $h(p_1, p_2)$ is the heightmap and h_0 is the specified minimum ground clearance, equal to $15m$ or $50m$ depending on how aggressive we want the UAV to fly. The objectives of the optimization are specified by

$$L(x, u) = k_1 ||u||^2 + k_2 p_3^2 \tag{6}$$

$$\Psi_{CTG}(x(t_0 + T)) = -k_3 p_2(t_0 + T) \tag{7}$$

where $k_1, k_2, k_3 > 0$. Thus, the trajectory cost penalizes controls as well as height while the cost-to-go makes the trajectory extend in the p_2 direction.

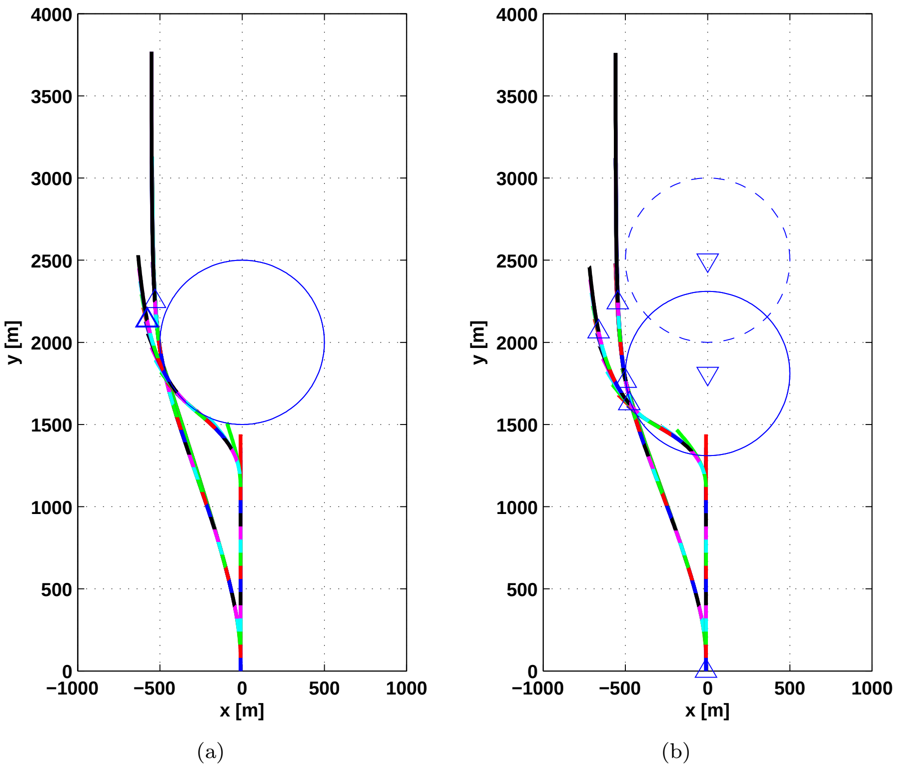

Figure 5. Comparison between standard RHC and GDS-RHC. Both algorithms have the same amount of nodes, but very different horizon lengths. In (a) a static obstacle is encountered, and in (b) another UAV moving at $30m/s$ is encountered. Both objects have a $500m$ safety distance. As can be seen, the longer planing horizon allows an earlier adaption in both cases, resulting in progress gaps of more than $100m$ after passing the obstacles. UAV positions at chosen time instants are shown by triangles.

First, in Figure 5, we illustrate the benefits of the longer planing horizon afforded by a GDS discretization compared to a regular RHC with the same number of nodes. The discretization depicted in Figure 4 was used and compared to a discretization where all five phases were of the same density as the first two in the GDS of the figure. The short term planning interval length T_1 was set to $T_1 = 1s$. In Figure 5(a) a static obstacle is to be avoided, and in Figure 5(b) the UAV encounter another UAV, heading in the opposite direction at a speed of $30m/s$. The problem parameters were the following $\|v\| = 80m/s$, $a_{\max} = 30m/s^2$, $k_1 = 1, k_2 = 0, k_3 = 10$ and $h(p_1, p_2) = 0$, essentially making it a planar problem. The obstacles $(q_1(t), q_2(t))$ where included in $g(x, u) < 0$ in terms of the constraint $||(p_1, p_2) - (q_1(t), q_2(t))|| > 500m$. In Figure 5(a), the triangles show the positions of both UAVs after $30s$ and it is clear that the longer horizon, afforded by the GDS, has resulted in faster progress towards the north. In Figure 5(b), the triangles show the positions of both UAVs after $0s$, $24s$, $30s$ as well as the encountered UAV at $0s, 24s$. As can be seen, both UAVs graze the $500m$ safety zone at $t = 24s$, and again, the one with the longer planning horizon (GDS) performs better. Note that while static obstacles can easily be included in the cost-to-go computation, moving obstacles can not, and the advantages of a long planning horizon is clear in both cases.

To illustrate how the density of the discretization is connected to the obstacle margins we ran the example with $\|v\| = 100m/s$, $a_{\max} = 6m/s^2$ and $k_1 = 14, k_2 = 0.45, k_3 = 3$. In Figure 3 above, one of the two trajectories is planned without extra margin in phase 3. As can be seen, that trajectory intersects a

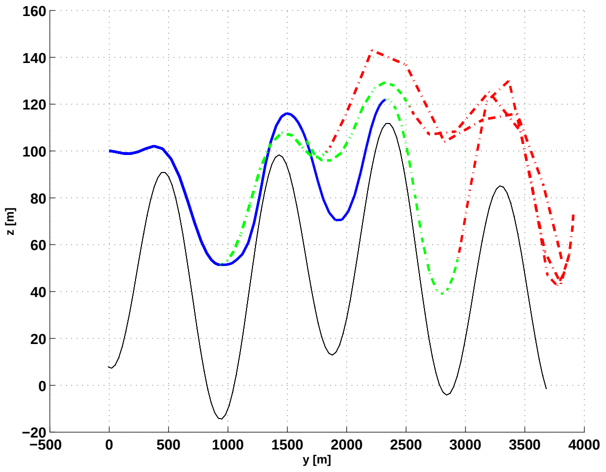

Figure 6. Three iterations of a three phase RHC. The first phase is drawn in solid blue, the second in dash-dotted green and the final in dash-dotted red. The first phases of each step is executed while the next iteration RHC is computed. Note how the large obstacle margins of the final phase are only have minor effects on the aggressiveness of the executed trajectory.

hill where the collocation points are very sparse. In the second trajectory the obstacle margins have been increased from $15m$ to $50m$, making the trajectory collision free. Note that there is only a small difference between the executed parts (solid blue) of the two trajectories. As explained in Section III above, there are a number of reasons for having larger margins in the later parts of the trajectory. Firstly, and as seen above, it might be needed to guarantee collision avoidance even when the collocation points are sparse. Secondly, it is reasonable to be more conservative due to uncertainties in measurements and motion of other vehicles.

To illustrate what effect large obstacle margins in the far future parts of the plan have on the performance of the executed trajectory, a three iteration long RHC scheme is shown in Figure 6. The trajectories are GDS, just as the ones in Figure 3, and we can see how the large margins of phase 2 and 3 are converted into aggressive phase 1 maneuvers, when the RHC iteration progresses.

Finally, we illustrate the behavior for both ordinary RHC and GDS in an example where the task is to pass a mountain ridge with a high altitude area. The difficulty is that, since altitude is penalized, once a trajectory has lead up to the ridge area, there are no short paths down to lower ground and it adds to the cost to continue at high altitude. (It is assumed that the grid on which a possible precomputed cost-to-go is defined is so coarse that detailed real time planning is needed to pass the area.) A Monte-Carlo simulation of this scenario with 100 samples is shown in Figure 7 and the statistics are summarized in Table 1. For

	RHC	GDS
Mean	-1.800×10^4	-2.407×10^4
Std. Dev.	3.847×10^3	1.774×10^3

Table 1. Statistics for the example in Figure 7.

each sample, the initial velocity in the x-direction is random and one trajectory of plans is computed for ordinary RHC as well as GDS. The parameters used are $p(0) = (0, 0, 70)m$, $v(0) = (v_r, 80, 0)m/s$ where v_r is a Gaussian random variable $v_r \sim N(0, 30)$, and $h_0 = 0m$, $a_{\max} = 30m/s^2$, $k_1 = 1, k_2 = 0.003, k_3 = 10$.

As can be seen, the GDS plans tend more often to find a cost effective route around the ridge, compared to RHC which more often tends to find a solution going over the ridge. This is due to the fact that over a short horizon, the instant progress of a climbing path in the y-direction outweighs the altitude penalty. Whereas over a longer horizon, the curved path around the ridge gives reasonable progress in the y-direction at a much lower altitude penalty. This indicates that the gradual dense sparse discretization serves its purpose; the sparsest parts are used to detect feasible economical paths stretching into the future which are then gradually refined until the moment when they are incorporated and executed in a detailed short term plan.

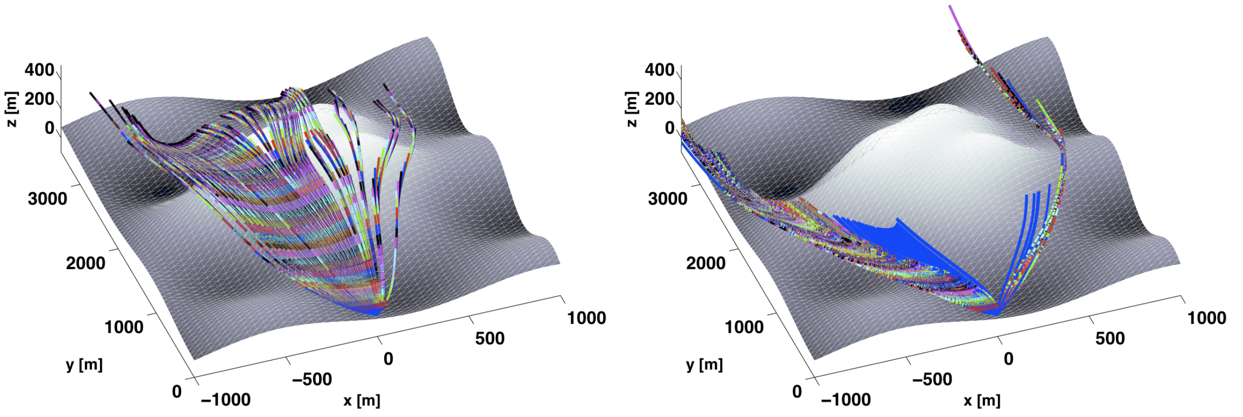

Figure 7. Comparison between ordinary RHC (left) and GDS (right) for varying initial condition. The initial velocity in the x-direction is random and for each random velocity one RHC trajectory and one GDS trajectory is calculated, consisting of 30 short term plans. The results from 100 independent trials are shown in each of the panels. It is clear that the GDS is more often capable of finding a more cost effective way around the mountain ridge instead of flying over it.

V. Conclusion

In this paper we have proposed a way of increasing the efficiency of online vehicle path planners using the direct Receding Horizon Control (RHC) approach. The basic idea is to adapt the allocation of computational resources to how the iterative plans are used. By using GDS discretizations, we make sure that the plans are detailed where they need to be, i.e., in the very near future, and less detailed further ahead.

The GDS discretizations can be exploited in two ways. Either fewer nodes are used, which leads to faster computations, or the same number of nodes are used, which leads to longer horizons which in turn improves performance.

The benefits of the GDS in terms of longer horizons were demonstrated in simulation examples. The increased horizon lengths gave a clear performance advantage when encountering both static and moving obstacles, as well as when flying low over terrain. The sparse far future parts of the plans enabled the vehicle using GDS-RHC to adopt to its environment much faster than the one using regular RHC.

Appendix

In GDS based on dyadic divisioning the RHC problem over $[t_0, t_0+T]$ is divided into $m+1$ phases, where the time duration of each phase is an even multiple of the duration T_1 of the short term plan and $T_1 = T/2^m$, for some positive integer m. All phases have a Legendre-Gauss node distribution for the collocation, with the same number of nodes, but the time duration of the phases gradually increase towards the end of the planning interval $[t_0, t_0+T]$ (so that the node density accordingly decreases).

A simple example of the idea is illustrated in Figure 8 for $m = 3$. Here, one starts with two phases of time duration T_1, followed by one of duration $2T_1$ and finally one of duration $4T_1$. In each of these phases the number of node points is one and the same; thus the node density (per time unit) is four times as high in the leftmost phase compared to the rightmost. As an initial condition, before the first pass of Algorithm 1, it is assumed that a plan has been computed over $[t_0, t_0+T]$ by solving Problem 2, where we take $t_0 = 0$ for simplicity. The leftmost phase of duration T_1 represents the short term plan with maximal node density and the following phases represent a gradual transition towards lower node density for longer planning times. During the first pass, which is performed during flight in $[0, T_1]$, a new phase of time duration $4T_1$ is appended and the end of the previous last phase and a new solution is computed over $[T_1, T+4T_1]$, reusing data from the previous plan as initial guess. In the second pass of Algorithm 1, which is performed during flight in the time interval $[T_1, 2T_1]$ the first phase where the node density is not already maximal, which is the one defined over $[2T_1, 4T_1]$, is split in half generating two new phases with maximal node density. To produce an initial guess for these two phases based on the old solution an Alpert wavelet reconstruction (encoding) step is used.[26] A new plan over $[2T_1, T+4T_1]$ is now computed based on the new set of phases. In the third pass,

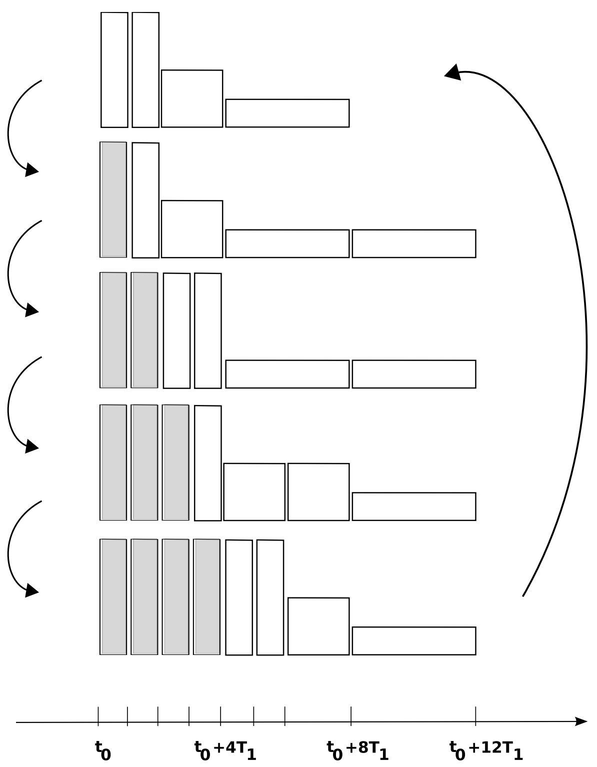

Figure 8. The GDS discretization with dyadic interval divisioning in a simple case with 4 phases and 3 different discretization densities. The node density in each phase is indicated by the box height which is inversely proportional to the box base length. Before the first pass of Algorithm 1 we have the situation at the top, and a plan has been computed over all four phases, together comprising the planning horizon $T = 8T_1$. At the first pass, a new phase of time duration $4T_1$ is added at the end and the plan is updated to hold also over this phase. This pass occurs during $[t_0, t_0 + T_1]$ and is indicated by the greyed first box in the second panel from the top. In the following three passes, each occurring during flight over a time interval of length T_1 (given by the rightmost box in the respective panel), the first phase from the left in the remaining planning phases which is not already at maximal node density is split in half using wavelet reconstruction, thus doubling the node density in the newly created phase, and the plan is updated accordingly reusing old data. With this scheme, the number of discretization nodes in the planning problem is kept constant in each pass.

which is performed during flight in $[2T_1, 3T_1]$, the first phase where the node density is not already maximal, which is the one defined over $[4T_1, 8T_1]$, is split in half using a wavelet reconstruction step, generating two new phases of half the maximal node density each. Again, a new plan is computed, now over $[3T_1, T + 4T_1]$, based on the new set of phases. In the fourth pass of Algorithm 1, which is performed during $[4T_1, 4T_1]$, the first phase where the node density is not maximal is split in half, which is the one defined over $[4T_1, 6T_1]$, and wavelet reconstruction is applied. After computation of a solution over $[4T_1, T + 4T_1]$ based on the new set of phases we are at the same situation as just before the first pass, and the process can be repeated.

The Alpert multiresolution framework[26] is based on locally defined polynomials on compact intervals that can be represented conveniently in terms of Lagrange interpolating polynomials using the Legendre-Gauss nodes. A key property of the framework is that the support of the wavelets on each resolution level is confined to either the left or right half of the support interval of the corresponding scaling functions. Together this means that the operation of phase splitting described above can be implemented efficiently using a reconstruction step of the Alpert multiresolution analysis since the Legendre-Gauss node distribution is the same as that used in the pseudo-spectral based solver used for the OCP problem.

The strategy outlined here keeps the amount of node (collocation) data held in memory constant in each pass and will reuse old plans in an efficient way to produce initial guesses. Moreover, the strategy can easily be generalized to more elaborate versions containing more dyadically divided phases with intermediate node densities.

References

[1]Dolgov, D., Thrun, S., Montemerlo, M., and Diebel, J., "Path planning for autonomous driving in unknown environments," *Experimental Robotics*, Springer, 2009.

[2]Urmson, C. et al., "Self-Driving Cars and the Urban Challenge," *IEEE Intelligent Systems*, 2008, pp. 66–68.

[3]Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., et al., "Junior: The Stanford entry in the Urban Challenge," *Journal of Field Robotics*, Vol. 25, No. 9, 2008.

[4]Baker, C., Ferguson, D., and Dolan, J., "Robust mission execution for autonomous urban driving," *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS)*, 2008.

[5]Kuwata, Y., Fiore, G., Teo, J., Frazzoli, E., and How, J., "Motion Planning for Urban Driving using RRT," *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*, 2008, pp. 1681–1686.

[6]Kuwata, Y. and How, J. P., "Three Dimensional Receding Horizon Control for UAVs," *AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, Rhode Island*, 16–19 August 2004.

[7]Mettler, B., "Combining On- and Offine Optimization Techniques for Effcient Autonomous Vehicle's Trajectory Planning," *AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco, California*, 15–18 August 2005.

[8]Anisi, D., Robinson, J. W. C., and Ogren, P., "Safe receding horizon control of an aerial vehicle," *Proc. of the 45 th IEEE Conference on Decision and Control, San Diego, CA*, 2006.

[9]Ögren, P. and Leonard, N. E., "A Convergent Dynamic Window Approach to Obstacle Avoidance," *IEEE Transactions on Robotics*, April 2005.

[10]Schouwenaars, T., How, J., and Feron, E., "Decentralized Cooperative Trajectory Planning of Multiple Aircraft with Hard Safety Guarantees," *AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, Rhode Island, August*, 2004.

[11]Schouwenaars, T., How, J., and Feron, E., "Receding horizon path planning with implicit safety guarantees," *Proc. of the IEEE American Control Conference*, Vol. 6, 2004, pp. 5576–5581.

[12]Richards, A., Kuwata, Y., and How, J. P., "Experimental Demonstrations of Real-time MILP Control," *AIAA Guidance, Navigation, and Control Conference and Exhibit, Austin, Texas*, 11–14 August 2003.

[13]Gong, Q., Kang, W., Bedrossian, N., Fahroo, F., Sekhavat, P., and Bollino, K., "Pseudospectral Optimal Control for Military and Industrial Applications," *IEEE Conference on Decision and Control*, 2007, pp. 4128–4142.

[14]Williams, P., "Application of pseudospectral methods for receding horizon control," *Journal of Guidance Control and Dynamics*, Vol. 27, No. 2, 2004, pp. 310–313.

[15]Huntington, G., *Advancement and Analysis of a Gauss Pseudospectral Transcription for Optimal Control Problems*, Ph.D. thesis, MIT, 2007.

[16]Benson, D., *A Gauss pseudospectral transcription for optimal control*, Ph.D. thesis, MIT, 2004.

[17]Stoer, J., Bulirsch, R., Gautschi, W., and Witzgall, C., *Introduction to numerical analysis*, Springer Verlag, 2002.

[18]Liseikin, V., *Grid generation methods*, Springer Verlag, 1999.

[19]Jain, S. and Tsiotras, P., "Sequential Multiresolution Trajectory Optimization for Moving Targets," *AIAA Guidance, Navigation, and Control Conference and Exhibit, Honolulu, Hawaii*, 2008.

[20]Anisi, D., "Adaptive node distribution for on-line trajectory planning," *Proceedings of the 25 th International Congress of the Aeronautical Sciences ICAS, Hamburg, Germany*, 2006.

[21]Binder, T., Blank, L., Dahmen, W., and Marquardt, W., "An Adaptive Multiresolution Method for Real-Time Moving Horizon Optimization," *Proc. American Control Conference, 2000*, Vol. 6, 2000, pp. 4234–4238.

[22]Kumar, R. and Seywald, H., "Dense-sparse discretization for optimization and real-time guidance," *Journal of guidance, control, and dynamics*, Vol. 19, No. 2, 1996, pp. 501–503.

[23]Canny, J., Donald, B., Reif, J., and Xavier, P., "On the complexity of kinodynamic planning," *Annual IEEE Symposium on Foundations of Computer Science*, Cornell University, 1988.

[24]Primbs, J., *Nonlinear Optimal control: A receding horizon approach*, Ph.D. thesis, California Institute of Technology, 1999.

[25]Alpert, B., "A class of bases in L^2 for the sparse representation of integral operators," *SIAM Journal on Mathematical Analysis*, Vol. 24, No. 1, 1993, pp. 246–262.

[26]Alpert, B., Beylkin, G., Gines, D., and Vozovoi, L., "Adaptive Solution of Partial Differential Equations in Multiwavelet Bases," *Journal of Computational Physics*, Vol. 182, No. 1, October 2002, pp. 149–190.

[27]Rao, A., Benson, D., Huntington, G., and Francolin, C., "Users manual for GPOPS, A MATLAB Package for Dynami Optimization Using the Gauss Pseudospectral Method," 2009.

[28]Dubins, L., "On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, Vol. 79, No. 3, 1957, pp. 497–516.