Efficient and More Accurate Representation of Solution Trajectories in Numerical Optimal Control

Yuanbo Nie and Eric C. Kerrigan *Senior Member, IEEE*

***Abstract*— We show via examples that, when solving optimal control problems, representing the optimal state and input trajectory directly using interpolation schemes may not be the best choice. Due to the lack of considerations for solution trajectories in-between collocation points, large errors may occur, posing risks if this solution is to be applied. A novel solution representation method is proposed, capable of yielding a solution of much higher accuracy for the same discretization mesh. This is achieved by minimizing the integral of the residual error for the overall trajectory, instead of forcing the errors to be zero only at collocation points. In this way, the requirement for mesh resolution can be significantly reduced, leaving the problem dimensions relatively small. This particular formulation also avoids some of the drawbacks found in the earlier work of integrated residual minimization, leading to more efficient computations.**

***Index Terms*— residual minimization, optimal control, solution representation, nonlinear predictive control**

I. Introduction

Solving optimal control problems (OCPs) are central in the field of trajectory optimization and real-time optimization-based control. In practice, most OCPs need to be solved with numerical schemes. In many situations, indirect methods can be difficult to implement, since they require analytic expressions of optimality conditions, which can be hard to derive. Direct methods have consequently become the standard for solving practical optimal control problems [1].

Direct methods often require transcription of the infinite-dimensional OCP into a nonlinear programming (NLP) problem of a finite dimension, via the introduction of a discretization mesh. If only the control trajectories are discretized as decision variables, the method is called sequential, of which direct single shooting is an example. In simultaneous approaches, such as direct multiple shooting and direct collocation, the state trajectories are also discretized and included as decision variables. A comprehensive comparison of different solution strategies are available in [2], [3], demonstrating the advantages of simultaneous methods in solving large-scale problems using sparse NLP formulations, compared to sequential methods.

Since the NLP solver will only return samples of the solution at a finite number of points in the discretization mesh, additional steps and care must be taken when representing the continuous-time results for time instances other than these sampled points. The common practice today is to directly use interpolation schemes, which are selected in accordance with the type of discretization mesh [4], [5].

Collocation methods with direct interpolation have a major drawback: the residual error of the ordinary differential equations (ODEs) is forced to zero only at collocation points. In general, no guarantees on accuracy and constraint satisfaction can be derived for system trajectories in-between collocation points. Posterior analysis is needed to identify intervals where errors are high and mesh refinement procedures are used to modify the discretization mesh. The problem has to be solved iteratively until all the errors are within tolerances.

We present a solution representation method that can significantly improve the solution accuracy for the same discretization mesh, so that results of higher quality are obtainable with relatively coarse meshes. This is achieved by minimizing the ODE residual error integrated over the solution trajectory. Section II provides the background information for solving optimal control problems numerically with the direct collocation method. Section III introduces the fundamental concept of the residual minimization method and motivates the development of the proposed scheme, which is presented in Section IV. The benefits of the method are demonstrated in Section V with two example problems, followed by concluding remarks in Section VI.

II. Numerical Optimal Control

Optimization-based control often requires the solution of OCPs expressed in the general Bolza form:

$$\min_{x,u,p,t_0,t_f} \Phi(x(t_0),t_0,x(t_f),t_f,p)+\int_{t_0}^{t_f} L(x(t),u(t),t,p)dt \tag{1a}$$

subject to

$$\dot{x}(t) = f(x(t),u(t),t,p),\ \forall t\in[t_0,t_f] \tag{1b}$$

$$c(x(t),u(t),t,p)\leq 0,\ \forall t\in[t_0,t_f] \tag{1c}$$

$$\phi(x(t_0),t_0,x(t_f),t_f,p)=0, \tag{1d}$$

with $x:\mathbb{R}\to\mathbb{R}^n$ is the state trajectory of the system, $u:\mathbb{R}\to\mathbb{R}^m$ is the control input trajectory, $p\in\mathbb{R}^s$ are static parameters, $t_0\in\mathbb{R}$ and $t_f\in\mathbb{R}$ are the initial and terminal time. Φ is the Mayer cost functional (Φ: $\mathbb{R}^n\times\mathbb{R}\times\mathbb{R}^n\times\mathbb{R}\times\mathbb{R}^s\to\mathbb{R}$), L is the Lagrange cost functional ($L:\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}\times\mathbb{R}^s\to\mathbb{R}$), f is the dynamic constraint ($f:\mathbb{R}^n\times$

Yuanbo Nie and Eric C. Kerrigan are with the Department of Aeronautics, Imperial College London, SW7 2AZ, U.K. yn15@ic.ac.uk, e.kerrigan@imperial.ac.uk

Eric C. Kerrigan is also with the Department of Electrical & Electronic Engineering, Imperial College London, London SW7 2AZ, U.K.

Accepted version to be published in: IEEE Control Systems Letters

©2019 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

$\mathbb{R}^m \times \mathbb{R} \times \mathbb{R}^s \to \mathbb{R}^n$), c is the path constraint ($c : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \times \mathbb{R}^s \to \mathbb{R}^{n_g}$) and ϕ is the boundary condition ($\phi : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^s \to \mathbb{R}^{n_q}$).

A. Direct collocation methods

Direct collocation methods can be categorized into fixed-order h methods (e.g. Euler, Trapezoidal, and Hermite-Simpson (H-S) as in [4]), and variable higher-order *p/hp* methods (e.g. Legendre-Gauss-Radau (LGR) as in [6]). Here, we aim to provide a high level overview. With a mesh of size $N := \sum_{k=1}^{K} N^{(k)}$, the states can be approximated as

$$x^{(k)}(\tau) \approx \bar{x}^{(k)}(\tau) := \sum_{j=1}^{N^{(k)}} \mathcal{X}_j^{(k)} \mathcal{B}_j^{(k)}(\tau), \tag{2}$$

within mesh interval $k \in \{1, \dots, K\}$, where $N^{(k)}$ is the number of collocation points for interval k, and $\mathcal{B}_j^{(k)}(\cdot)$ are basis functions. For classical h methods, $\tau \in \mathbb{R}^N$ takes on values in the interval $[0, 1]$ representing $[t_0, t_f]$, and $\mathcal{B}_j^{(k)}(\cdot)$ are chosen to be elementary B-splines of various orders. For *p/hp* methods, $\mathcal{B}_j^{(k)}(\cdot)$ are Lagrange interpolating polynomials over the normalized time interval $\tau \in [-1, 1]$. We use $X_j^{(k)}$ and $U_j^{(k)}$ to represent the approximated states and inputs at collocation points, e.g. $X_j^{(k)} = \bar{x}^{(k)}(\tau_j^{(k)}) \in \mathbb{R}^n$, where $\tau_j^{(k)}$ is the j^{th} collocation point in mesh interval k.

Consequently, the OCP (1) can be approximated by

$$J_c := \min_{X,U,p,t_0,t_f} \Phi(X_1^{(1)}, t_0, X_f^{(K)}, t_f, p) + \sum_{k=1}^{K} \sum_{i=1}^{N^{(k)}} w_i^{(k)} L(X_i^{(k)}, U_i^{(k)}, \tau_i^{(k)}, t_0, t_f, p) \tag{3a}$$

subject to, for $i = 1, \dots, N^{(k)}$ and $k = 1, \dots, K$:

$$\sum_{j=1}^{N^{(k)}} \mathcal{A}_{ij}^{(k)} X_j^{(k)} + \mathcal{D}_i^{(k)} f(X_i^{(k)}, U_i^{(k)}, \tau_i^{(k)}, t_0, t_f, p) = 0 \tag{3b}$$

$$c(X_i^{(k)}, U_i^{(k)}, \tau_i^{(k)}, t_0, t_f, p) \leq 0 \tag{3c}$$

$$\phi(X_1^{(1)}, t_0, X_f^{(K)}, t_f, p) = 0 \tag{3d}$$

where $w_j^{(k)}$ are the quadrature weights for the respective discretization method chosen, $\mathcal{A}$ is the numerical differentiation matrix with $\mathcal{A}_{ij}$ the element (i, j) of the matrix, and $\mathcal{D}$ a constant matrix. The discretized problem can then be solved with off-the-shelf or structure-exploiting NLP solvers.

B. Representing the results

The NLP solver generates a discretized solution $\mathcal{Z} := (X, U, p, t_0, t_f)$ as sampled data points. Interpolating splines may be used to construct an approximation of the continuous-time solution $\tilde{z}(t) := (\tilde{x}(\mathcal{Z}, t), \tilde{u}(\mathcal{Z}, t), t, p)$, with $\tilde{x}(\mathcal{Z}, \cdot)$, $\tilde{u}(\mathcal{Z}, \cdot)$ the approximated state and input trajectories.

1) Representation via direct interpolation: Conventionally, the interpolation of the solution corresponds to the discretization scheme used in the transcription process. Thus, we must analyze how the state approximation (2) enters the optimal control problem formulation (3a).

It is not difficult to discover that the only dependency on the basis function in (3a) appears in the defect constraint (3b), through the first term representing the numerical differentiation of the approximated function $\bar{x}(\cdot)$.

For most commonly-used numerical schemes, the numerical differentiation formulation has an equivalent integration form. Both forms are presented in Table I, where $h_k := \Delta t(\tau_N^{(k)} - \tau_1^{(k)})$, $\Delta t := t_f - t_0$, and

$$F_i^{(k)} := f(X_i^{(k)}, U_i^{(k)}, \tau_i^{(k)}, t_0, t_f, p). \tag{4}$$

For each numerical scheme, direct interpolation of the OCP solution is possible using splines with the type and order in accordance with Table II. For example, with Hermite-Simpson transcription, the reconstructed state trajectory inside mesh interval k using cubic splines will be

$$\begin{aligned}\tilde{x}^{(k)}(\mathcal{Z}, t) = {} & X_1^{(k)} + F_1^{(k)}(t - t_1^{(k)}) \\ & + \frac{1}{2}\left(-3F_1^{(k)} + 4F_2^{(k)} - F_3^{(k)} \right) \frac{(t - t_1^{(k)})^2}{h_k} \\ & + \frac{2}{3}\left(F_1^{(k)} - 2F_2^{(k)} + F_3^{(k)} \right) \frac{(t - t_1^{(k)})^3}{h_k^2},\end{aligned} \tag{5}$$

the dynamics trajectory with quadratic splines will be

$$\begin{aligned}\dot{\tilde{x}}^{(k)}(\mathcal{Z}, t) = {} & F_1^{(k)} + \left(-3F_1^{(k)} + 4F_2^{(k)} - F_3^{(k)} \right) \frac{t - t_1^{(k)}}{h_k} \\ & + \left(2F_1^{(k)} - 4F_2^{(k)} + 2F_3^{(k)} \right) \left(\frac{t - t_1^{(k)}}{h_k} \right)^2,\end{aligned} \tag{6}$$

TABLE I
TYPICAL NUMERICAL SCHEMES

Method (Order)	**Numerical Integration Scheme**
Euler (1)	$X_2^{(k)} = X_1^{(k)} + h_k F_1^{(k)}$
Trapezoidal (2)	$X_2^{(k)} = X_1^{(k)} + \frac{h_k}{2}(F_1^{(k)} + F_2^{(k)})$
Hermite Simpson (3)	$X_2^{(k)} = \frac{1}{2}(X_2^{(k)} + X_1^{(k)}) + \frac{h_k}{8}(F_1^{(k)} - F_3^{(k)})$ $X_3^{(k)} = X_1^{(k)} + \frac{h_k}{6}(F_1^{(k)} + 4F_2^{(k)} + F_3^{(k)})$
LGR ($N^{(k)}$)	$\mathcal{I}^{(k)} = [\mathcal{A}_{2:N+1}^{(k)}]^{-1}$ $X_{2:N+1}^{(k)} = X_1 + \frac{\Delta t}{2}\mathcal{I}^{(k)} F_{1:N}^{(k)}$

TABLE II
CONTINUITY OF THE RECONSTRUCTED SOLUTION (A.M.: AT MOST; P.W.: PIECEWISE)

Method	**Dynamics ($\dot{\tilde{x}}$)**	**States ($\tilde{x}$)**	**Inputs ($\tilde{u}$)**
Euler	p.w. constant	a.m.p.w. linear	same as dynamics
Trape.	a.m.p.w. linear	a.m.p.w. quad.	
H-S	a.m.p.w. quad.	a.m.p.w. cubic	
LGR	a.m. order $N^{(k)}$	a.m. order $N^{(k)}$+1	

and the control trajectory with quadratic splines will have the expression

$$\tilde{u}^{(k)}(\mathcal{Z},t) = \frac{2}{h_k^2}(t-\frac{1}{2}t_1^{(k)}-\frac{1}{2}t_3^{(k)})(t-t_3^{(k)})U_1^{(k)} - \frac{4}{h_k^2}(t-t_1^{(k)})(t-t_3^{(k)})U_2^{(k)} + \frac{2}{h_k^2}(t-t_1^{(k)})(t-\frac{1}{2}t_1^{(k)}-\frac{1}{2}t_3^{(k)})U_3^{(k)}. \quad (7)$$

for all $t \in [t_1^{(k)}, t_3^{(k)}]$. The whole trajectory $\tilde{x}(\mathcal{Z},\cdot)$, $\dot{\tilde{x}}(\mathcal{Z},\cdot)$ and $\tilde{u}(\mathcal{Z},\cdot)$ can then be expressed as piecewise polynomials.

For p/hp methods, Lagrange interpolating polynomials are often used as basis functions during the transcription process. An alternative version, namely barycentric Lagrange interpolation, is often used instead for solution interpolation, due to its improved numerical stability.

2) Evaluation of errors: The quality of the interpolated solution needs to be assured through error analysis, assessing the level of accuracy and constraint satisfaction. Firstly, any valid trajectory $\tilde{z}(\cdot)$ must satisfy the system dynamics (1b) with a good level of accuracy. Therefore, one measure for the error due to discretization and interpolation is through the calculation of the ODE residual $\varepsilon_r(t) \in \mathbb{R}^n$ defined as

$$\varepsilon_r(t) := \dot{\tilde{x}}(\mathcal{Z},t) - f(\tilde{x}(\mathcal{Z},t), \tilde{u}(\mathcal{Z},t), t, p). \quad (8)$$

For the discretized problem, the error in the state variables over each interval in-between collocation points can then be estimated with the integral

$$\eta_j := \int_{t_j}^{t_{j+1}} \|\varepsilon_r(s)\|_2 \, ds,$$

as a single metric for a multi-variable problem, or

$$\sigma_{j,q} := \int_{t_j}^{t_{j+1}} |\varepsilon_{r_q}(s)| \, ds, \text{ for } q = 1,\ldots,n,$$

for each dynamics equation separately. $\eta \in \mathbb{R}^N$ or $\sigma \in \mathbb{R}^{n\times N}$ are typically referred to as the *absolute local error* [4]. The operator $\|\cdot\|_2$ is the vector 2-norm. The integral can be practically estimated by high order quadrature.

In addition, numerical discretization inevitably leads to possible constraint violations of the trajectories in-between the collocation points. For path and box constraints that are expressed semi-explicitly as (1c), the *absolute local constraint violation* $\varepsilon_{c_\zeta}(t) \in \mathbb{R}^{n_g}$ may be straight-forwardly estimated by

$$\varepsilon_{c_\zeta}(t) := \begin{cases} 0 & \text{if } c_\zeta(\tilde{z}(t)) \le 0 \\ c_\zeta(\tilde{z}(t)) & \text{if } c_\zeta(\tilde{z}(t)) > 0 \end{cases}, \text{ for } \zeta = 1,\ldots,n_g.$$

Once the distributions of errors are calculated, appropriate modifications can be made to the discretization mesh, to iteratively resolve the problem until the obtained solution fulfills all predefined error tolerances (η_{tol} and $\varepsilon_{c_{tol}}$). This process is called mesh refinement (MR). Common approaches for mesh refinement include adding intervals and/or changing the polynomial order. The NLP formulated based on the new mesh is warm started using the previous solution from the coarser mesh. This can often lead to significantly faster convergence, compared to a fine uniform mesh without MR, thus reducing the overall computation time.

C. Problems associated with direct reconstruction

Practical experience has shown that trajectory interpolation in accordance with the discretization scheme is not the best choice. In many cases large discretization errors and constraint violations occur inside the intervals in-between collocation points. Furthermore, if the optimal control trajectory is discontinuous, direct interpolation using polynomials can often result in a Gibbs-like phenomenon, inducing non-physical oscillations in the solution. A typical example for this to happen would be in problems with bang-bang control, if the switch happens in-between mesh intervals.

These issues are fundamentally rooted in the direct collocation formulation. Firstly, states, dynamics and controls can rarely all be approximated accurately by polynomials. Even in the simple case where $f(x(t),u(t)) = \dot{x}(t) = ax(t)+u(t)$ and $u(t) = 1$ are both polynomials (thus can be represented exactly by polynomials), the corresponding state trajectory $x(t) = x(0)e^{at} + \int_0^t e^{a(t-s)}u(s)\ ds$ is clearly not a polynomial and approximation errors should be expected.

It is then important to note that driving the defect constraint (3b) to zero (or machine precision) at collocation points does not imply that the polynomial functions used for the state and input approximations in the NLP will satisfy the dynamic equations and constraints in-between collocation points. In fact, the opposite can and often does occur.

It is well-known in the field of curve fitting that if a function cannot be exactly represented by a polynomial, forcing the polynomial to exactly go though some sampled data points generally results in larger errors in comparison to fitting using least squares criteria. The same analogy can be applied here: forcing the defect constraints to be zero at collocation points will generally result in larger overall defect errors for the whole trajectory, in comparison to a method that minimizes the integral of the defect errors in a least squares manner. This observation motivated the development of the integrated residual minimization method.

III. Method of integrated residual minimization

Integrated residual minimization is motivated by the recently-proposed method in [7], which is a generalization of the least-squares approach for solving differential equations to solving dynamic optimization problems. The idea is that instead of forcing the ODE residuals (8) to be zero at collocation points with (3b), the method tries to minimize the square of the 2-norm of the ODE residuals for the represented solution polynomials integrated along the whole trajectory, i.e.

$$\min_{\hat{x},\hat{u},t,p} \int_{t_0}^{t_f} r(\hat{x}(t),\hat{u}(t),t,p)\ dt \quad (9)$$

with

$$r(\hat{x}(t),\hat{u}(t),t,p) := \|\dot{\hat{x}}(t) - f(\hat{x}(t),\hat{u}(t),t,p)\|_2^2. \quad (10)$$

As presented in [7], the expressions for state and input functions $\hat{x}$ and $\hat{u}$ can be polynomials of any standard types, with polynomial coefficients $P_{j,q}$ as decision variables.

This concept shares some resemblance with the method of direct error enforcement in [8], where the residual errors are computed with a different formulation and only at a single non-collocation reference point for each mesh interval.

Remark 1: The choice of representation in [7] with $P_{j,q}$ as decision variables increases the computational complexity of the problem in comparison to direct collocation:

- One extra decision variable is required for every state and input variable in every mesh segment.
- Simple bounds need to be implemented as general inequality constraints.
- Additional computations are required to obtain the initial guesses of the decision variables from an estimation of the solution trajectory.
- The magnitudes of decision variables may span a wide numerical range. This is detrimental in terms of ensuring consistent numerical accuracy in computations.
- If finite differences are used for obtaining the derivative information, the calculations can be less accurate.
- Proper scaling of decision variables can be difficult.
- State continuity in-between mesh segments might need to be enforced with additional equality constraints.

Therefore, we need to develop a method that avoids the above-listed drawbacks and, to a great extent, retains the computational efficiency of direct collocation.

IV. The proposed scheme

Based on the above observations, we propose a method to generate solution trajectories that can be orders of magnitude more accurate than direct interpolation for the ODE defect error, without increasing the size of the discretization mesh. The method retains the same decision variables as in (3), namely $\mathcal{Z} := (X, U, p, t_0, t_f)$, and uses the interpolation polynomial formula $\tilde{x}(\mathcal{Z}, \cdot)$, $\dot{\tilde{x}}(\mathcal{Z}, \cdot)$ and $\tilde{u}(\mathcal{Z}, \cdot)$ to directly represent $\hat{x}(\cdot)$, $\dot{\hat{x}}(\cdot)$ and $\hat{u}(\cdot)$ in (9) and (10).

For example, consider Hermite-Simpson discretization. The input trajectory inside mesh interval k can be represented by the polynomial as in (7), based on the values of the decision variables $U_1^{(k)}$, $U_2^{(k)}$ and $U_3^{(k)}$. However, for the state trajectory, one challenge arises. For solutions to (3), continuity of state variables are automatically fulfilled when using (5) as the interpolation equation; however, this is not generally the case for arbitrary solutions that violate the defect constraint (3b).

To avoid imposing additional path constraints for state continuity, we make use of the original Hermite-Simpson numerical integration scheme (in Table I), and obtain

$$F_2^{(k)} = -\frac{1}{2h_k}(5X_1^{(k)} - 4X_2^{(k)} - X_3^{(k)} + F_1^{(k)}h_k) \quad (11)$$

$$F_3^{(k)} = \frac{1}{h_k}(4X_1^{(k)} - 8X_2^{(k)} + 4X_3^{(k)} + F_1^{(k)}h_k). \quad (12)$$

As a check, substituting $t = t_2^{(k)} = t_1^{(k)} + h_k/2$ and (11) into (5) will result in $X_2^{(k)}$, and substituting $t = t_3^{(k)} = t_1^{(k)} + h_k$ and (12) into (5) will result in $X_3^{(k)}$. Thus, with $F_1^{(k)}$, $F_2^{(k)}$ and $F_3^{(k)}$ calculated based on (4), (11) and (12), respectively, the interpolation formula (5) guarantees state trajectory continuity without imposing additional constraints.

Thus, an optimization problem for representing the OCP solution can be formulated as

$$\min_{X,U,p,t_0,t_f} \sum_{k=1}^{K} R(X^{(k)}, U^{(k)}, \tau^{(k)}, \tau_q^{(k)}, t_0, t_f, p) \quad (13a)$$

subject to, for $i = 1, \ldots, N^{(k)}$ and $k = 1, \ldots, K$,

$$\begin{aligned} \sum_{k=1}^{K}\sum_{i=1}^{N^{(k)}} w_i^{(k)} L(X_i^{(k)}, U_i^{(k)}, \tau_i^{(k)}, t_0, t_f, p) \\ +\Phi(X_1^{(1)}, t_0, X_f^{(K)}, t_f, p) \leq J_c \end{aligned} \quad (13b)$$

$$c(X_i^{(k)}, U_i^{(k)}, \tau_i^{(k)}, t_0, t_f, p) \leq 0 \quad (13c)$$

$$\phi(X_1^{(1)}, t_0, X_f^{(K)}, t_f, p) = 0 \quad (13d)$$

with (13b) the constraint for the objective of the original OCP (3) and $J_c \in \mathbb{R}$ the value of the cost obtained from direct collocation. R is the residual cost: for certain problems, this can be calculated precisely with analytical expressions; for most practical problems, quadrature rules of sufficiently high order can be used, i.e.

$$\begin{aligned} &R(X^{(k)}, U^{(k)}, \tau^{(k)}, \tau_q^{(k)}, t_0, t_f, p) := \\ &\sum_{\iota=1}^{N_q^{(k)}} w_\iota^{(k)} \|\dot{\tilde{x}}(\mathcal{Z}, t_{q_\iota}^{(k)}) - f(\tilde{x}(\mathcal{Z}, t_{q_\iota}^{(k)}), \tilde{u}(\mathcal{Z}, t_{q_\iota}^{(k)}), t_{q_\iota}^{(k)}, p)\|_2^2 \end{aligned}$$

with $t_{q_\iota}^{(k)} := \frac{t_f^{(k)} - t_0^{(k)}}{2}\tau_{q_\iota}^{(k)} + \frac{t_f^{(k)} + t_0^{(k)}}{2}$ and $\tau_q^{(k)} \in \mathbb{R}^{N_q^{(k)}}$ the quadrature mesh for approximating the integral inside a mesh interval, where $w_\iota^{(k)}$ are the corresponding quadrature weights. Typically a Gaussian quadrature of order $N_q^{(k)} \geq 4N^{(k)} + 1$ is required for a good accuracy [7].

When comparing (13) to the original OCP formulation of direct collocation (3), it is straightforward to note that, if $\mathcal{Z}$ is a solution to (3), then $\mathcal{Z}$ will also be a feasible point for (13). If no better feasible solution is obtainable, (13) is at least guaranteed to have one solution, namely $\mathcal{Z}$.

In terms of computational complexity, the proposed formulation (13) avoids the shortcomings as listed in Remark 1. In addition, unlike the penalty-barrier finite element (PBF) method proposed in [7], which requires tailored solvers for a good performance, (13) can be efficiently solved with the same off-the-shelf sparse NLP solvers as direct collocation. The transcription and the majority of the computational components can be shared between the two, and warm starting techniques can be exploited to accelerate the computations.

V. Example Problems

Here, we present two example problems to demonstrate the main advantages of the proposed scheme. Both OCPs are transcribed using the optimal control software `ICLOCS2` [9], and numerically solved to a tolerance of 10^{-9} with NLP solver `IPOPT` [10] (version 3.12.9). With extremely coarse

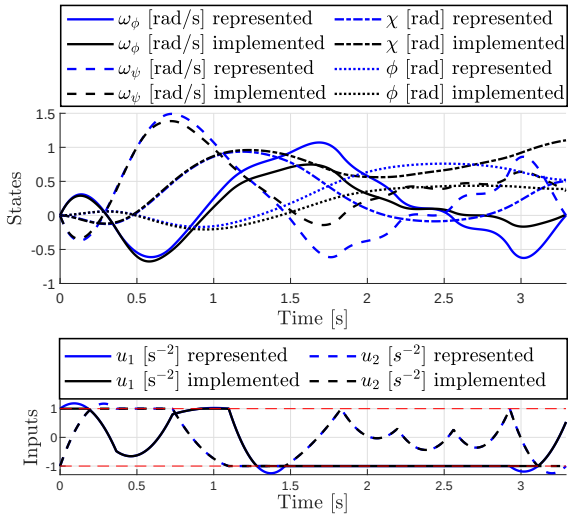


Fig. 1. Solution to the two-link robot arm problem, direct interpolation method for solution representation, direct collocation with Hermite-Simpson discretization, 10 mesh intervals

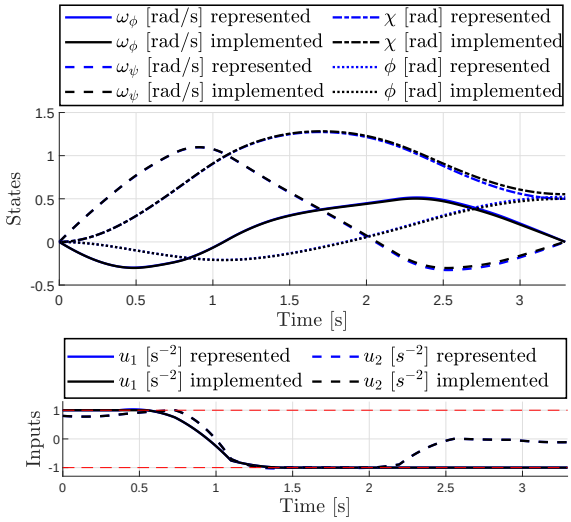


Fig. 2. Solution to the two-link robot arm problem, integrated residual minimization method for solution representation, direct collocation with Hermite-Simpson discretization, 10 mesh intervals

meshes, the emphasis of the comparison will not be on yielding solutions that look similar to the optimal trajectory. Instead, the goal is to obtain sub-optimal solutions that, when applied, can result in low discrepancies between the represented solution and the implementation outcome.

A. *Two-Link Robot Arm*

The two-link robot arm problem presented here was adapted from [11, Ex. 2, Sect. 12.4.2]. Consider a system consisting of two identical beams with the same property (mass: $m = 1\,\text{kg}$, length: $l = 1\,\text{m}$, and moment of inertia), connected at two actuated joints. The objective is to reposition a payload of mass $M = 1\,\text{kg}$ in minimum time, with the addition of a regularization term:

$$\min_{x,u,t_f} \quad t_f + 0.01 \int_0^{t_f} u_1(t)^2 + u_2(t)^2 \, dt.$$

The system has angular rates ω_ϕ, ω_ψ, and angles ϕ, $\chi = \phi - \psi$ as state variables, and nondimensionalized torque u_1 and u_2 as inputs. Furthermore, the variable simple bounds and boundary conditions are imposed in accordance to the reference, except that $\chi(t_f) = 0.5\,\text{rad}$, and $\phi(t_f) = 0.522\,\text{rad}$.

Figures 1 and 2 illustrate the solutions to the two-link robot arm problem problem generated with the two different solution representation methods. Presented alongside are the outcomes from the actual implementation of the resultant input trajectory on the same dynamic model, solved with a non-stiff variable-order ODE solver (Matlab `ode113`) with a time step 100 times smaller than the discretization grid of the optimization problem. Observe that:

- Despite a very small tolerance and successful termination of the NLP solver, the collocation solution and interpolation of the solution exhibit large errors, leading to significant deviations to the state trajectories when the inputs are directly applied. In contrast, only minor discrepancies can be observed for the solutions represented using integrated residual minimization, on the same coarse grid with relatively low-order discretization.
- Although the constraints are implemented in the exact same way, the proposed method, to a greater extent, alleviates the issues of constraint violations inside the mesh intervals. This is because these constraint violations are often related to the large ODE defect errors in-between collocation points, which are directly dealt with by the residual minimization scheme.

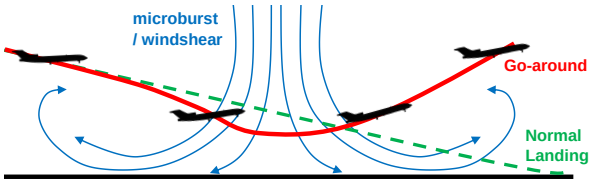


Fig. 3. Illustration of the aircraft go-around in windshear problem

B. *Aircraft Go-around in the Presence of Windshear*

Based on previous developments [12], a problem is presented in [4] where the aircraft needs to stay as high above the ground as possible after encountering a severe windshear during landing. See the illustration in Figure 3.

A simplified windshear model is used with wind speed contributions represented by a horizontal and a vertical component. Other details about the aerodynamic modelling, parameter values, simple bounds and boundary conditions are the same as in [4]. A static parameter h_{min} is introduced to represent the minimum altitude. The objective is therefore to minimize $-h_{min}$ together with path constraint $h(t) \geq h_{min}$.

The angle of attack α is the actual control input to the physical system (aircraft); however, in order to implement

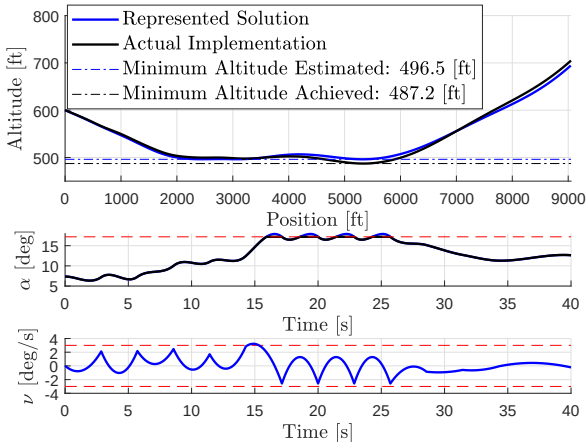


Fig. 4. Solution to the aircraft go-around in the windshear problem, direct interpolation method for solution representation, direct collocation with Hermite-Simpson discretization, 15 mesh intervals

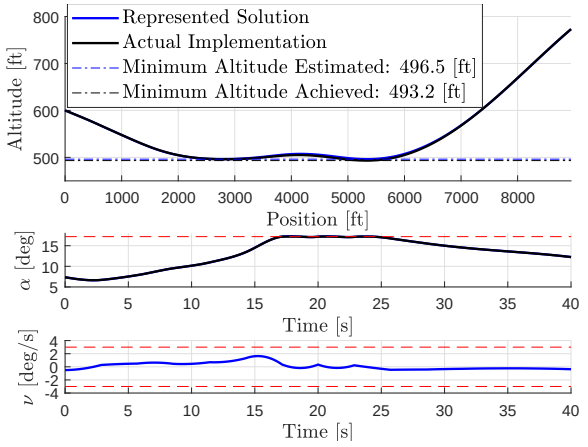


Fig. 5. Solution to the aircraft go-around in the windshear problem, integrated residual minimization method for solution representation, direct collocation with Hermite-Simpson discretization, 15 mesh intervals

a constraint on its rate of change, ν is introduced as angle of attack rate and serves as the control input with $\dot{\alpha}(t) = \nu(t)$. This implementation is known to exhibit singular arc behaviour [4], [13], [14], leading to fluctuations and ringing phenomena in the solutions.

The solutions to this problem are collectively shown in Figures 4 and 5. In addition to the advantages identified in the previous example, solution representation via integrated residual minimization have clear benefits in suppressing fluctuations. This ringing phenomenon is frequently observed in direct collocation solutions of singular control problems, as well as the directly interpolated solution trajectories.

VI. Conclusions

Although interpolation of collocation solutions can sometimes yield good results, it is very difficult to guarantee accuracy without posterior error assessments and mesh design iterations. As shown by the examples, despite successfully solving the NLP to negligibly small tolerance, the validity of the solution may still be questionable with large discrepancies. The flaws are rooted in collocation schemes, where the ODE defect errors are forced to zero at collocation points, regardless of the errors inside the intervals.

The proposed solution representation method of integrated residual minimization fundamentally addresses this shortcoming by instead minimizing the integrated ODE residual error along the whole trajectory. As a result, solutions of higher accuracy are obtainable with the same discretization mesh, allowing the mesh to be relatively coarse. This benefit is clearly demonstrated with the example problems: despite being highly nonlinear, moderately complex and solved on a coarse low-order mesh, only minor differences are observed between the represented solution and actual implementation.

Since solving OCPs with numerical methods are essentially multi-objective optimization problems, one will inevitably face the trade-off between minimizing the objective (for optimality) and minimizing the discretization errors (for accuracy). By utilizing the objective value from a collocation solution, our proposed approach is yet to offer complete flexibility for the user in managing this trade-off process. Further development of this concept into a standalone method could potentially offer more benefits.

References

[1] D. J. N. Limebeer and A. V. Rao, "Faster, higher, and greener: vehicular optimal control," *IEEE Control Systems*, vol. 35, no. 2, pp. 36–56, 2015.

[2] A. Cervantes and L. T. Biegler, "Optimization strategies for dynamic systems," *Encyclopedia of optimization*, pp. 1886–1897, 2001.

[3] L. T. Biegler, "An overview of simultaneous strategies for dynamic optimization," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1043–1053, 2007.

[4] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming: Second Edition*. Advances in Design and Control, Society for Industrial and Applied Mathematics, 2010.

[5] M. Kelly, "An introduction to trajectory optimization: how to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017.

[6] F. Liu, W. W. Hager, and A. V. Rao, "An hp mesh refinement method for optimal control using discontinuity detection and mesh size reduction," in *53th IEEE Conference on Decision and Control (CDC)*, pp. 5868–5873, IEEE, 2014.

[7] M. P. Neuenhofen and E. C. Kerrigan, "Dynamic optimization with convergence guarantees," *arXiv preprint arXiv:1810.04059*, 2018.

[8] S. Vasantharajan and L. T. Biegler, "Simultaneous strategies for optimization of differential-algebraic systems with enforcement of error criteria," *Computers & Chemical Engineering*, vol. 14, no. 10, pp. 1083–1100, 1990.

[9] Y. Nie, O. J. Faqir, and E. C. Kerrigan, "ICLOCS2: Solve your optimal control problems with less pain," in *Proc. 6th IFAC Conference on Nonlinear Model Predictive Control*, 2018.

[10] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[11] R. Luus, *Iterative Dynamic Programming*. Monographs and Surveys in Pure and Applied Mathematics, Taylor & Francis, 2000.

[12] A. Miele, T. Wang, H. Wang, and W. Melvin, "Optimal penetration landing trajectories in the presence of windshear," *Journal of Optimization Theory and Applications*, vol. 57, no. 1, pp. 1–40, 1988.

[13] Y. Nie and E. C. Kerrigan, "How should rate constraints be implemented in nonlinear optimal control solvers?," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 362–367, 2018.

[14] Y. Nie and E. C. Kerrigan, "Efficient implementation of rate constraints for nonlinear optimal control," *arXiv preprint arXiv:1903.05508*, 2019.