2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

A Hybrid Quadratic Programming Framework for Real-Time Embedded Safety-Critical Control

Ryan M. Bena, Sushmit Hossain, Buyun Chen, Wei Wu, and Quan Nguyen

***Abstract*— We present a new framework for implementing real-time embedded safety-critical controllers which utilizes hybrid computing to address the issue of limited computational resources, a problem that is particularly prevalent in microrobotics. In our approach, the nominal stabilizing control algorithm is implemented digitally while the safety-critical quadratic program is solved via a dedicated analog resistor array. We apply this hybrid computing architecture to a simulated collision avoidance task for a micro-aerial vehicle and show the benefit relative to a purely-digital implementation. By leveraging analog quadratic programming on the Crazyflie 2.1 micro quadrotor, a reduction in overall processing time from 8.9 ms to 0.6 ms is estimated for this computationally-limited system. We further display the viability of our proposed safety-critical control framework through real-time flight demonstrations, utilizing a novel prototype analog circuit tethered to the Crazyflie. The flight results confirm the functionality of the control structure and prototype circuit while highlighting the overall capabilities of hybrid computing.**

I. INTRODUCTION

Analog electronics have been used to execute embedded feedback control algorithms since the 1940s when developments in radar technologies motivated a need for precision tracking that could be automated using analog servomechanisms [1], [2]. During the following decades, analog circuits became increasingly prevalent in real-time controller implementation, quickly prompting discussion of the impact of discrete sampling on stability [3]. In recent years, many embedded controllers have moved away from analog implementations into the digital domain. As a result, state-of-the-art controllers are primarily implemented via software on microprocessors, field programmable gate arrays, or dedicated *microcontrollers* (MCUs). Devices such as these are often ideal for computing simple arithmetic operations: the basis of most classical and modern feedback control algorithms. For instance, the widely-used classical technique of *proportional-derivative* (PD) control and the state-space formulation of state feedback both employ static gains to scale actuated control effort based on calculated error values, thus achieving system stability [4]. Many traditional nonlinear control techniques similarly utilize arithmetic functions to determine appropriate control actions [5]. The prevalence of these approaches in modern autonomy can be directly attributed to their simplicity, both theoretically and computationally, which allows for a straightforward digital implementation.

The authors are with the Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90089, USA (bena@usc.edu; hossains@usc.edu; buyunche@usc.edu; wwu808@usc.edu; quann@usc.edu)

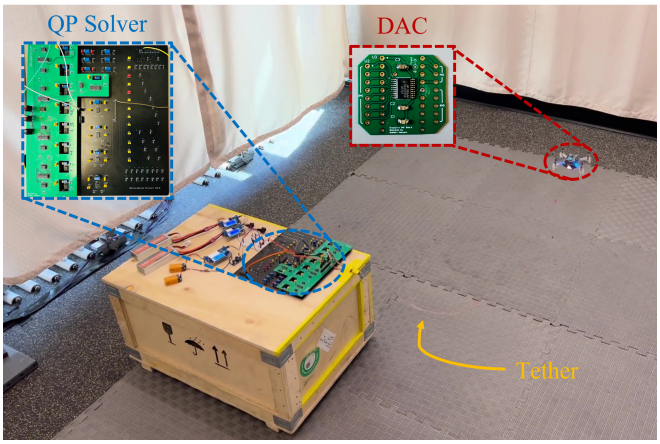

Fig. 1: **Experimental Flight - Hybrid Quadratic Programming.** A safety-critical flight demonstration using a novel analog QP solver circuit tethered to a Crazyflie 2.1 micro quadrotor with limited onboard computational capabilities. Video available at https://youtu.be/mL6m9I_TClw.

The emerging field of safety-critical control addresses the need for autonomous system architectures which simultaneously meet stability and safety criteria. *Model predictive control* (MPC) is one approach which has repeatedly demonstrated its effectiveness in this arena by dynamically addressing system state and input constraints [6]. A more modular way of achieving safety objectives is through the implementation of an optimal safety-critical filter which enforces *control barrier function* (CBF), *control Lyapunov function* (CLF), and physical control-input constraints [7]. The performance benefits of this method have been highlighted in [8]–[10]. Due to its effectiveness and reduced computational complexity in comparison to MPC, CBF-based safety-critical control is the technique employed herein.

One shared characteristic of the aforementioned safety-critical control techniques is the incorporation of quadratic programming, a problem formulation which sets out to minimize a quadratic objective function subject to linear equality and inequality constraints, enabling online control optimization. *Quadratic programs* (QPs) are traditionally solved with numerical search algorithms which iteratively approach the optimal solution [11]. Unfortunately, these algorithms are computationally expensive, limiting their utility for real-time embedded controllers, especially on micro-scale systems, such as the Crazyflie 2.1, which frequently operate at high bandwidths with minimal digital processing resources. Due to this limitation, safety-critical control algorithms for microrobotic systems are generally processed off-board, and the resultant control command is wirelessly relayed to the robot [12]–[14].

979-8-3503-2365-8/23/$31.00 ©2023 IEEE

Hybrid computing is a viable solution to this problem. Drawing inspiration from biology and neuroscience, hybrid controllers can split digital and analog computing tasks along similar lines as the human cerebrum and cerebellum [15]. The digital domain (cerebrum) handles complex tasks such as perception and decision making; meanwhile, the analog domain (cerebellum) controls movement and stability. In our previous work, we leveraged this concept to balance a mobile robot using an analog PD controller and an analog sensor fusion circuit [16]. In the case of safety-critical control, researchers have proven that analog resistor arrays can efficiently compute QP solutions with a high degree of accuracy [17]–[19]. Other optimization circuits using analog elements have been recently proposed in [20]–[25].

In this paper, we present a safety-critical control framework that enables the first integration of an analog QP solver into a real-time control application, thus bringing closer to reality the possibility of incorporating QP-based control architectures onboard microrobotic systems with fast dynamics and computationally-limited MCUs. The details of our approach will be expounded as follows: Section II presents background on CBF-based safety-critical control theory and analog QP-solving resistor arrays; Section III describes our hybrid safety-critical control architecture utilizing analog quadratic programming, Section IV shows simulation results comparing a baseline digital controller to the proposed hybrid controller when applied to a *micro-aerial vehicle* (MAV), Section V assesses relevant experimental results using a Crazyflie 2.1 MAV tethered to a custom analog circuit, and Section VI provides the overall conclusions of this research.

II. PRELIMINARIES

A. Safety-Critical Control: Control Barrier Functions

The primary objective of safety-critical controller synthesis is to design a feedback law which achieves system tracking objectives while also enforcing situationally-dependent safety restrictions. The desired tracking task is characterized by a state reference trajectory, $\boldsymbol{x}_{\text{d}} \in \mathbb{R}^n$, and the safety requirements take the form of a *safe-set*, $\mathcal{C}$, i.e. a limit on the value of the system state, $\boldsymbol{x} \in \mathbb{R}^n$, represented by

$$\mathcal{C} = \{\boldsymbol{x} \in \mathbb{R}^n : B(\boldsymbol{x}) \geq 0\}, \tag{1}$$

where $B : \mathbb{R}^n \mapsto \mathbb{R}$ is a continuously differentiable function defining the boundary of the safe-set.

CBFs are a tool by which, for affine control systems whose dynamics evolve according to

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{\mu}, \tag{2}$$

the state constraint in (1) can be enforced by way of a dynamic linear constraint with respect to the control input, $\boldsymbol{\mu} \in \mathbb{R}^m$ [26].

For a safe-set of *relative degree* N, exponential CBF constraints take the form

$$B^{(N)}(\boldsymbol{x}, \boldsymbol{\mu}) + \cdots + K_1\dot{B}(\boldsymbol{x}) + K_0 B(\boldsymbol{x}) \geq 0, \tag{3}$$

where, for $i \in \{0, 1, \ldots, N-1\}$, $K_i \in \mathbb{R}$ denotes a user-defined coefficient. Adherence to (3) guarantees convergence of the system state to the safe-set in (1). A detailed derivation of (3), as well as the requirements on all coefficients, K_i, to guarantee convergence, can be found in [27].

Because the system dynamics are affine and the safe set has relative degree N, the control first appears linearly in $B^{(N)}(\boldsymbol{x}, \boldsymbol{\mu})$; therefore, the inequality in (3) can be rearranged to form

$$\boldsymbol{A}_{\text{cbf}}(\boldsymbol{x})\boldsymbol{\mu} \leq b_{\text{cbf}}(\boldsymbol{x}), \tag{4}$$

where $\boldsymbol{A}_{\text{cbf}} : \mathbb{R}^n \mapsto \mathbb{R}^{1\times m}$ and $b_{\text{cbf}} : \mathbb{R}^n \mapsto \mathbb{R}$ represent the parameters of the final linear inequality constraint.

Given a nominal, or desired, control action, $\boldsymbol{\mu}_{\text{d}} \in \mathbb{R}^m$, generated by some traditional control approach (e.g. state feedback), the goal of CBF-based safety-critical control is to determine the minimum deviation, $\boldsymbol{\mu}_{\text{qp}} \in \mathbb{R}^m$, which, when added to $\boldsymbol{\mu}_{\text{d}}$, produces a final control input,

$$\boldsymbol{\mu} = \boldsymbol{\mu}_{\text{d}} + \boldsymbol{\mu}_{\text{qp}}, \tag{5}$$

that satisfies (4). Furthermore, the final control, $\boldsymbol{\mu}$, should be physically achievable by the plant with minimal destabilizing effect on the tracking control task. This combination of requirements can be expressed via a QP of the form:

CLF-CBF-QP

$$\begin{aligned} \underset{\boldsymbol{\mu}_{\text{qp}}}{\arg\min} \quad & \frac{1}{2}\boldsymbol{\mu}_{\text{qp}}^T \boldsymbol{H}_{\text{qp}} \boldsymbol{\mu}_{\text{qp}} + \frac{1}{2}\rho\delta^2 \\ \text{subject to} \quad & \boldsymbol{A}_{\text{cbf}}(\boldsymbol{x})\boldsymbol{\mu} \leq b_{\text{cbf}}(\boldsymbol{x}), \\ & \boldsymbol{A}_{\text{clf}}(\boldsymbol{x})\boldsymbol{\mu} \leq b_{\text{clf}}(\boldsymbol{x}) + \delta, \\ & \boldsymbol{\mu}_{\min} \leq \boldsymbol{\mu} \leq \boldsymbol{\mu}_{\max}, \\ & \delta \geq 0. \end{aligned} \tag{6}$$

In this QP, $\boldsymbol{H}_{\text{qp}} \in \mathbb{R}^{m\times m}$ is a positive diagonal matrix and $\rho \in \mathbb{R}$ is a positive scalar. The slack variable, $\delta \in \mathbb{R}$, softens the CLF constraint defined by $\boldsymbol{A}_{\text{clf}} : \mathbb{R}^n \mapsto \mathbb{R}^{1\times m}$ and $b_{\text{clf}} : \mathbb{R}^n \mapsto \mathbb{R}$. The CLF constraint parameters can be derived from a valid CLF [28]. Upper and lower bounds on the control input are captured in $\boldsymbol{\mu}_{\min} \in \mathbb{R}^m$ and $\boldsymbol{\mu}_{\max} \in \mathbb{R}^m$.

QPs of the form in (6) do not generally have an analytical solution; therefore, various numerical algorithms have been developed which can approximate the optimal solution. The most widely used QP-solving algorithms for this application involve interior point methods [29]. These algorithms start with an initial feasible solution to the QP and incrementally adjust that solution along a calculated search direction. The search process is repeated iteratively until a user-defined stopping criterion is reached. The stopping criterion is usually based on a maximum iteration count or desired solution accuracy. One important aspect of interior point methods, and other numerical QP-solving algorithms, is that they often require extensive computation and/or many iterations to converge to an accurate solution, and this complexity can result in relatively slow convergence times, from a real-time safety-critical control perspective.

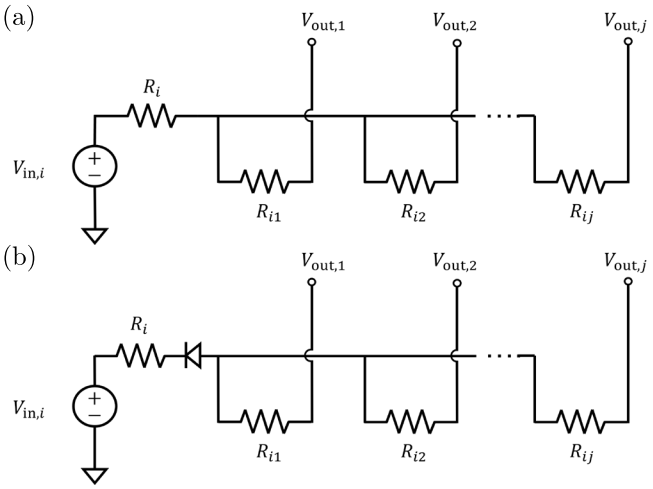

Fig. 2: **Analog QP Solver Modules.** a) Equality constraints are represented by parallel resistors connected to a shared DC voltage source. b) Inequality constraints are represented by a similar circuit, but with an ideal diode connected in series with R_i to ensure unidirectional current flow.

B. Analog Quadratic Programming

Consider a QP of the form:

Quadratic Program

$$\begin{aligned} \arg\min_{\boldsymbol{z}} \quad & \frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} \\ \text{subject to} \quad & \boldsymbol{A}_{\text{ineq}}\boldsymbol{z} \leq \boldsymbol{b}_{\text{ineq}}, \\ & \boldsymbol{A}_{\text{eq}}\boldsymbol{z} = \boldsymbol{b}_{\text{eq}}, \end{aligned} \tag{7}$$

where $\boldsymbol{z} \in \mathbb{R}^s$ is the optimization parameter, $\boldsymbol{H} \in \mathbb{R}^{s\times s}$ is a positive definite Hessian matrix, $\boldsymbol{A}_{\text{ineq}} \in \mathbb{R}^{p\times s}$ and $\boldsymbol{b}_{\text{ineq}} \in \mathbb{R}^p$ define p inequality constraints, and $\boldsymbol{A}_{\text{eq}} \in \mathbb{R}^{q\times s}$ and $\boldsymbol{b}_{\text{eq}} \in \mathbb{R}^q$ define q equality constraints. By inspection, we see that the CLF-CBF-QP in (6) has this general form.

From the resistor-based analog quadratic programming theory presented in [17]–[19], Fig. 2 shows the circuit modules needed to construct a solver for (7). Specifically, the circuits in Fig. 2(a) and Fig. 2(b) represent each equality and inequality constraint, respectively. By connecting the output nodes of these modules in an $r \times s$ array, where $r = p + q$ is the total number of constraints, an analog circuit can be constructed which satisfies all the linear constraints of a QP.

For each constraint, the resistances, R_{ij}, are calculated by

$$R_{ij} = \frac{1}{a_{ij}}, \tag{8}$$

where a_{ij} is the (i, j) element of either $\boldsymbol{A}_{\text{eq}}$ or $\boldsymbol{A}_{\text{ineq}}$. The negative resistor, R_i, must have the precise value

$$R_i = -\frac{1}{\Sigma_j a_{ij}}, \tag{9}$$

and the voltage applied to each input node is

$$V_{\text{in},i} = \frac{b_i}{\Sigma_j a_{ij}}, \tag{10}$$

where b_i is the i^{th} component of either $\boldsymbol{b}_{\text{eq}}$ or $\boldsymbol{b}_{\text{ineq}}$.

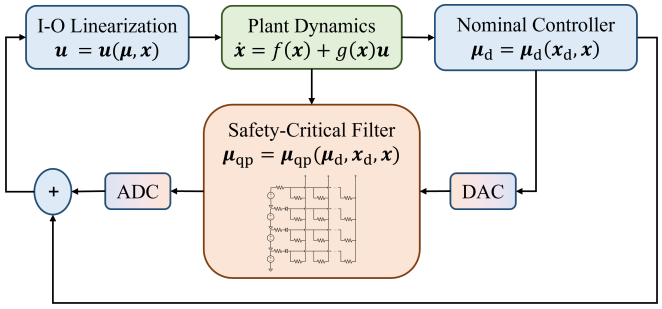

Fig. 3: **General Hybrid Control Architecture.** The structure of a hybrid analog/digital controller for any dynamic plant (green). The controller uses analog circuitry (orange) to solve the QP in the safety-critical filter and digital arithmetic (blue) to calculate the remainder of the control algorithm.

The voltages measured at the output nodes, $V_{\text{out},j}$, of the analog QP array will settle to steady state values corresponding to the elements of the QP optimization parameter, $\boldsymbol{z} = [z_1 \ z_2 \ \ldots \ z_n]^T$, and satisfying the prescribed linear equality and inequality constraints.

However, the measured value of $\boldsymbol{z}$ does not inherently minimize the objective function in (7). Instead, the array naturally minimizes a quadratic objective function with a positive definite Hessian matrix, $\bar{\boldsymbol{H}} \in \mathbb{R}^{s\times s}$, define by

$$\bar{\boldsymbol{H}} = \text{diag}(\boldsymbol{1}^T\boldsymbol{A}) - \boldsymbol{A}^T\text{diag}(\boldsymbol{1}^T\boldsymbol{A}^T)^{-1}\boldsymbol{A}, \tag{11}$$

where $\boldsymbol{A} \in \mathbb{R}^{r\times s}$ is the concatenated matrix of equality and inequality constraints, $\boldsymbol{A} = [\boldsymbol{A}_{\text{eq}}^T \ \boldsymbol{A}_{\text{ineq}}^T]^T$, and $\boldsymbol{1}$ is a column vector of ones with appropriate dimension.

By adding additional inactive constraints of the form $\boldsymbol{A}_{\text{i}}\boldsymbol{z} \leq \infty$ to the analog QP array, the natural Hessian matrix, $\bar{\boldsymbol{H}}$, can be manipulated to match the designed Hessian, $\boldsymbol{H}$, with arbitrarily small error [18]. After this step, the resultant output voltages of the QP array will represent the minimizing solution, $\boldsymbol{z}^* \in \mathbb{R}^s$.

III. HYBRID CONTROL ARCHITECTURE

In principal, the control architecture in Section II-A is sufficient to achieve position control objectives while adhering to stability, safety, and saturation constraints. However, numerical QP solution methods are slow and computationally expensive. In contrast, many real-time controllers, such as those in microrobotic systems, must be executed at high frequencies (100 - 1000 Hz) using embedded MCUs with limited computing power. This incompatibility reduces the viability of quadratic programming as a control methodology for such applications. The following proposed framework, depicted in Fig. 3, mitigates the issue by eliminating digital QP solvers and replacing them with analog resistor arrays.

The first step in our hybrid computing approach is to apply *input-output* (I-O) linearization to the plant dynamics [30]. I-O linearization is inherently model-dependent, so an accurate description of the nonlinear dynamics is necessary to ensure robust closed-loop behavior. Subsequently, a nominal controller is applied to the I-O linearized system, followed by an analog safety-critical filter which solves the QP in (6). To compute the CLF-CBF-QP solution using analog

electronics, the constraints must first be transformed into a suitable form. Precisely, the QP must be modified such that all elements of the concatenated inequality constraint matrix, $\boldsymbol{A}_{\text{ineq}}$, are greater than or equal to zero, thus ensuring that the resistor values calculated in (8) are all non-negative. This transformation is achieved by creating a new optimization parameter, $\boldsymbol{\nu}_{\text{qp}} \in \mathbb{R}^{2s}$, which contains the original optimization parameters, $\boldsymbol{\mu}_{\text{qp}}$ and δ, and their negatives, via the equality constraint

$$\left[\begin{array}{cc} \boldsymbol{\mathcal{I}} & \boldsymbol{\mathcal{I}} \end{array}\right] \boldsymbol{\nu}_{\text{qp}} = \mathbf{0}, \tag{12}$$

where $\boldsymbol{\mathcal{I}}$ is an identity matrix of appropriate dimension. Next, the new inequality constraint, $\bar{\boldsymbol{A}}_{\text{ineq}} \in \mathbb{R}^{p \times 2s}$, is formed by

$$\bar{\boldsymbol{A}}_{\text{ineq}} = \left[\begin{array}{cc} \boldsymbol{A}_{\text{ineq}}^{+} & \boldsymbol{A}_{\text{ineq}}^{-} \end{array}\right], \tag{13}$$

where the $(\cdot)^{+}$ and $(\cdot)^{-}$ operators maintain the absolute values of the positive and negative terms of the argument matrix, respectively, and set the remaining terms to zero.

Finally, the natural cost of the linear constraints, computed using (11), is driven to approximate the Hessian of the CLF-CBF-QP objective function in (6) by adding another variable, $\gamma \in \mathbb{R}$, to the optimization parameter, such that

$$\left[\begin{array}{cc} \mathbf{0} & 1 \end{array}\right] \left[\begin{array}{c} \boldsymbol{\nu}_{\text{qp}} \\ \gamma \end{array}\right] = 0, \tag{14}$$

along with the inactive inequality constraints

$$\left[\begin{array}{ccc} \boldsymbol{\Delta} & \mathbf{0} & \boldsymbol{d} \\ \mathbf{0} & \boldsymbol{\Delta} & \boldsymbol{d} \end{array}\right] \left[\begin{array}{c} \boldsymbol{\nu}_{\text{qp}} \\ \gamma \end{array}\right] \leq \boldsymbol{\infty}, \tag{15}$$

where $\boldsymbol{d} \in \mathbb{R}^{s}$ is a vector whose elements additively scale the diagonal elements of the natural cost matrix, $\boldsymbol{\Delta} \in \mathbb{R}^{s \times s}$ is defined by $\boldsymbol{\Delta} = \text{diag}(\boldsymbol{d})$, and $\boldsymbol{\infty}$ is a vector of appropriate dimension whose elements are all infinite. The values of $\boldsymbol{d}$ are chosen to be large relative to the values of the original natural cost Hessian, thus forcing the natural cost to approximate the cost in (6) with arbitrarily small error.

Resultantly, the transformed QP constraints become

$$\begin{gathered} \left[\begin{array}{ccc} \boldsymbol{A}_{\text{ineq}}^{+} & \boldsymbol{A}_{\text{ineq}}^{-} & \mathbf{0} \\ \boldsymbol{\Delta} & \mathbf{0} & \boldsymbol{d} \\ \mathbf{0} & \boldsymbol{\Delta} & \boldsymbol{d} \end{array}\right] \left[\begin{array}{c} \boldsymbol{\nu}_{\text{qp}} \\ \gamma \end{array}\right] \leq \left[\begin{array}{c} \boldsymbol{b}_{\text{ineq}} \\ \boldsymbol{\infty} \end{array}\right], \\ \left[\begin{array}{ccc} \boldsymbol{\mathcal{I}} & \boldsymbol{\mathcal{I}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1 \end{array}\right] \left[\begin{array}{c} \boldsymbol{\nu}_{\text{qp}} \\ \gamma \end{array}\right] = \mathbf{0}. \end{gathered} \tag{16}$$

From these constraints, the value of all necessary analog signals can be computed using (8)-(10), and the corresponding analog resistor array can be constructed.

One final aspect of integrating this analog QP-solver with embedded digital electronics is the incorporation of *digital-to-analog* (D/A) and *analog-to-digital* (A/D) conversion. In the proposed framework, all control calculations leading up to the computation of (16) are executed digitally. Once the parameters for the safety-critical filter QP have been fully determined, the corresponding voltages and resistances are computed, and the voltages are sent to the analog QP circuit

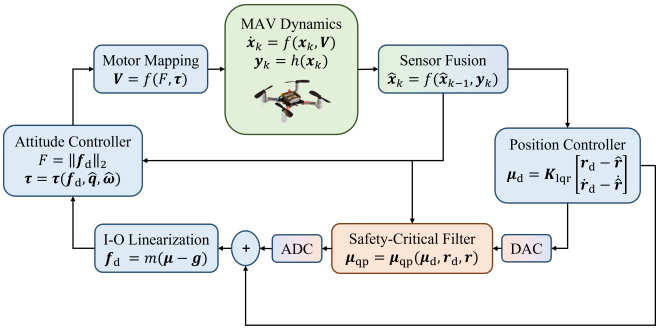

Fig. 4: **MAV Simulation Architecture.** The structure of a hybrid analog/digital controller for an MAV plant (green). The controller uses analog circuitry (orange) to solve the optimizing QP filter and digital arithmetic (blue) to calculate the remainder of the control algorithm.

using a *D/A converter* (DAC). The resistance values can also be updated in real-time using a variety of possible solutions including digital potentiometers, variable resistors, and/or memristors [16]. After the analog QP circuit reaches steady-state, the output voltages are measured using an *A/D converter* (ADC). These voltages represent $\boldsymbol{\nu}_{\text{qp}}$ and γ, and the first m terms constitute $\boldsymbol{\mu}_{\text{qp}}$. The remainder of the control architecture is implemented digitally, and the necessary commands are sent to the system actuators.

IV. SIMULATION

The architecture presented in Section III was applied to a quadrotor MAV model to simulate safety-critical flight on a system with limited digital resources (systems like this can benefit the most from hybrid computing). Motivated by the quadrotor collision avoidance methodologies presented in [14], [31], [32], our control framework in Fig. 3 was customized for a real-time MAV implementation as detailed in the following paragraphs and depicted in Fig. 4.

At the start of the control loop, the position reference, $\boldsymbol{r}_{\text{d}} \in \mathbb{R}^3$, the associated translation state estimate, $\hat{\boldsymbol{r}} \in \mathbb{R}^3$, and their derivatives, were sent to a nominal position control algorithm which used feedback based on the *Linear Quadratic Regulator* (LQR) algorithm as follows:

$$\boldsymbol{\mu}_{\text{d}} = \boldsymbol{K}_{\text{lqr}} \left[\begin{array}{c} \boldsymbol{r}_{\text{d}} - \hat{\boldsymbol{r}} \\ \dot{\boldsymbol{r}}_{\text{d}} - \hat{\dot{\boldsymbol{r}}} \end{array}\right], \tag{17}$$

where the output is a virtual acceleration vector, $\boldsymbol{\mu}_{\text{d}} \in \mathbb{R}^3$, representing the desired closed-loop acceleration of the MAV, and $\boldsymbol{K}_{\text{lqr}} \in \mathbb{R}^{3 \times 6}$ is the LQR state-feedback gain matrix.

The virtual acceleration vector, $\boldsymbol{\mu}_{\text{d}}$, was passed through the safety-critical filter, based on (6), which determined $\boldsymbol{\mu}_{\text{qp}}$. The safe-set for each dynamic obstacle was defined by

$$B(\boldsymbol{r}) = \frac{(r_x - r_{\text{c},x})^2}{a^2} + \frac{(r_y - r_{\text{c},y})^2}{b^2} + \frac{(r_z - r_{\text{c},z})^2}{c^2} - D^2, \tag{18}$$

which represents an ellipsoid at position $\boldsymbol{r}_{\text{c}} \in \mathbb{R}^3$ with semi-axes defined by $\{a, b, c\} \in \mathbb{R}$ and nominal barrier distance $D \in \mathbb{R}$. The input boundary parameters were calculated based on MAV acceleration limits, and the CLF constraint was designed to match the nominal LQR controller.

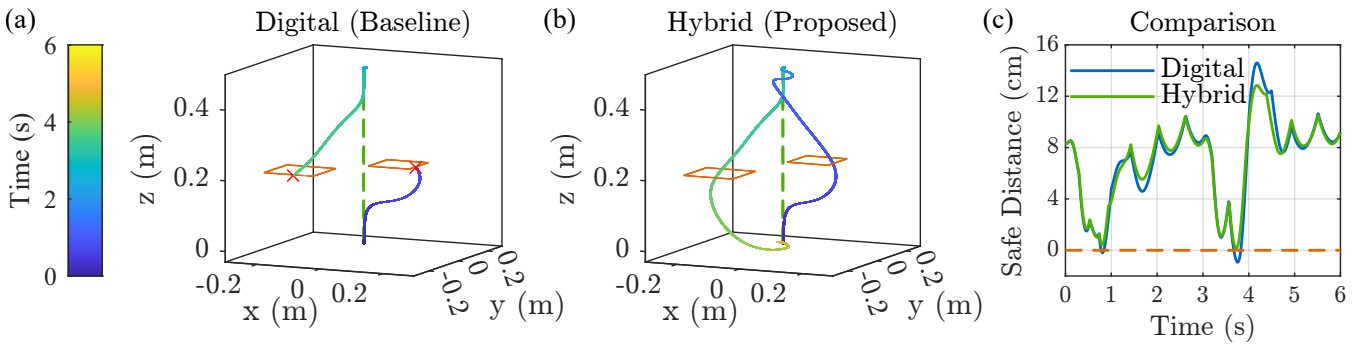

Fig. 5: **Simulation Results - MAV Collision Avoidance.** a) The flight path and reference (green) for a 100 Hz position control loop frequency, corresponding to a baseline digital computing approach. The clearance window between the four dynamic obstacles (orange) is depicted at the two points of closest approach. The flight resulted in collisions during ascent and descent. b) The flight path for a 1000 Hz position control loop frequency, corresponding to our proposed hybrid computing framework. The flight was collision-free. c) The distance between the MAV and the closest obstacle for both simulations.

Once $\boldsymbol{\mu}$ was calculated, the desired force vector, $\boldsymbol{f}_{\mathrm{d}} \in \mathbb{R}^3$, was computed using using the I-O linearization

$$\boldsymbol{f}_{\mathrm{d}} = m\left(\boldsymbol{\mu} - \boldsymbol{g}\right), \tag{19}$$

where $m \in \mathbb{R}$ is the MAV mass, and $\boldsymbol{g} = [0 \quad 0 \quad -9.81]^T$ $\mathrm{m/s^2}$ is Earth's gravitational acceleration vector.

To determine appropriate position control loop frequencies, we estimated the total computational time associated with both the digital and hybrid safety-critical controllers, assuming comparable solution errors. The purely-digital solution time was measured using an STM32F405 MCU onboard a Crazyflie 2.1 MAV, with the digital QP optimization accomplished via a custom `CVXGEN` solver. The hybrid solution time is a composite estimate which includes: all requisite digital computations, DAC communication at 21 MHz, digital resistor programming, analog QP settling, and ADC read-out. The final computation-time estimates, and their resultant position control loop frequencies, are presented in Table I. These control loop rates represent the maximum implementable frequencies on the Crazyflie, based on the corresponding computational time.

TABLE I: SAFETY-CRITICAL CONTROL FREQUENCIES

Method	Computational Time (ms)	Position Control Loop (Hz)
Digital	8.9	100
Hybrid	0.6	1000

Up to this point, the control algorithm leverages the assumption that the associated MAV translational dynamics are fully-actuated. However, because quadrotors can only apply thrust along the body-vertical axis, the system is actually underactuated, and therefore $\boldsymbol{f}_{\mathrm{d}}$ cannot be applied directly. Instead, $||\boldsymbol{f}_{\mathrm{d}}||_2$ dictates the thrust magnitude, while the vector direction is supplied as a reference to an attitude control subsystem operating at 1000 Hz. The attitude control algorithm assumes fully-actuated rotational dynamics. With attitude and angular velocity feedback, the attitude controller computes the torque required to align the body-vertical axis of the MAV with $\boldsymbol{f}_{\mathrm{d}}$ using the quaternion-based nonlinear algorithm presented in [33]. The thrust magnitude and attitude control torques are mapped to four independently controlled motors, steering the quadrotor MAV to follow the desired position trajectory while satisfying safety-critical constraints.

The control architecture was implemented on a closed-loop dynamic system model of the Crazyflie using Matlab Simulink. All digital quadratic programming was accomplished with `quadprog`. For the simulated flights, the MAV was commanded to track a vertical ascent, hover, and descent trajectory while avoiding four ellipsoidal dynamic obstacles. The time-varying trajectories of the obstacles were designed such that the obstacle boundaries overlapped, and the only safe MAV trajectory required passing through a narrow clearance window, demanding ideal controller performance to avoid a collision. The results are shown in Fig. 5 with animations provided in the Supplemental Material.

Fig. 5 illustrates that, in the established scenario, the MAV collides with the obstacles on both ascent and descent when operating with a baseline 100 Hz position control loop frequency. However, when the frequency is increased to 1000 Hz, corresponding to our proposed hybrid computing framework, the MAV successfully passes through the clearance window while avoiding contact with all four dynamic obstacles. This simulation highlights the direct impact of control frequency on safety-critical performance. Namely, safe-set convergence via (6) is only guaranteed in continuous time. As such, for microrobotic systems, or other systems with limited computational resources, dynamic collision avoidance tasks necessitate fast controllers, which can most effectively be realized with our hybrid safety-critical control architecture.

V. EXPERIMENTAL RESULTS

We demonstrated our hybrid QP framework on a Crazyflie 2.1 MAV in order to verify feasibility and functionality. First, we developed the prototype analog QP solver *printed circuit board* (PCB) depicted in Fig. 1. The negative resistance values in this PCB were realized with negative impedance converters, while the positive resistance values were produced by ceramic resistors and potentiometers. This PCB represents a novel contribution to the field of analog quadratic programming and is the first such resistor array to be used in a real-time control environment. The Crazyflie was tethered to the analog QP-solver prototype using a bundle

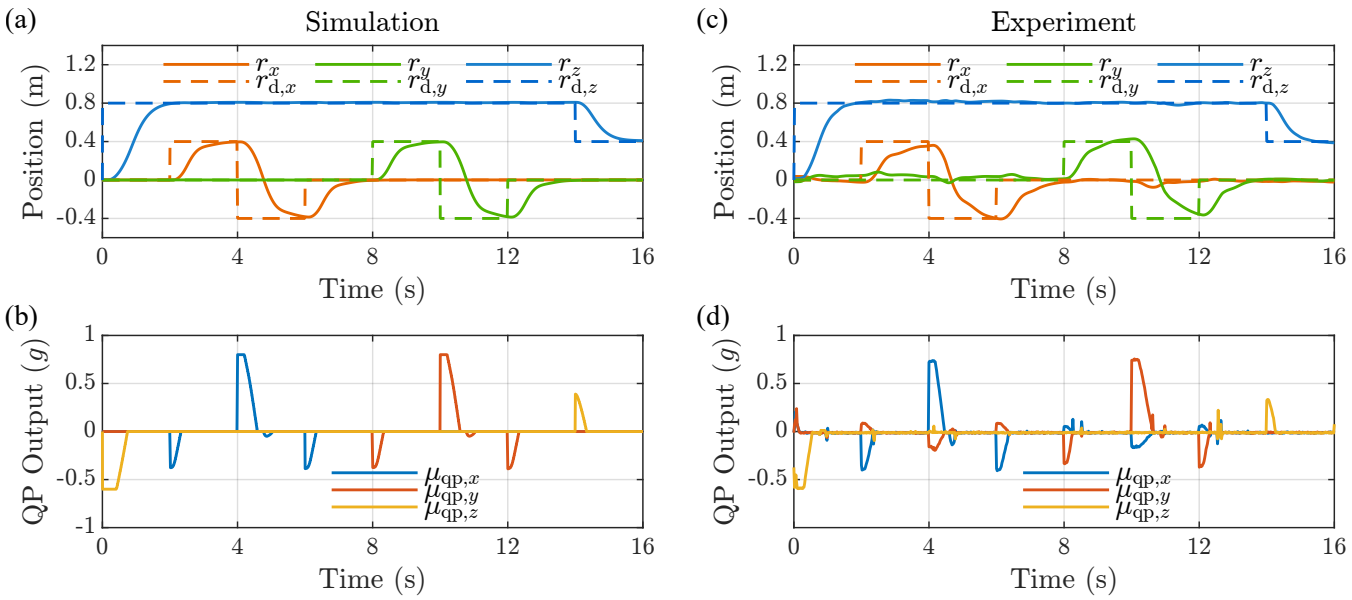

Fig. 6: **Demonstration Results - Hybrid Control Saturation.** a) The simulated position, $\boldsymbol{r}$, and corresponding reference trajectory, $\boldsymbol{r}_{\mathrm{d}}$, of an MAV during a flight with the safety-critical filter. b) The QP solution, $\boldsymbol{\mu}_{\mathrm{qp}}$, of the safety-critical filter during the simulated flight. c) The measured position, $\boldsymbol{r}$, and corresponding reference trajectory, $\boldsymbol{r}_{\mathrm{d}}$, of the Crazyflie 2.1 during a real-time flight experiment with the tethered analog safety-critical filter. d) The measured QP solution, $\boldsymbol{\mu}_{\mathrm{qp}}$, of the analog circuit during the tethered flight.

of 44-AWG enamel coated copper wires. The inputs to the analog PCB were generated by a MAX5725 DAC which was attached to the MAV using the custom-built breakout board also shown in Fig. 1. The analog PCB outputs were measured using the Crazyflie MCU's built-in ADC.

The safety constraints in (6) are dynamic; meanwhile, due to the prototypical nature of our analog circuit, none of the resistive elements could be adjusted during flight. Therefore, the dynamic CLF/CBF constraints were disconnected, and only the static input constraints remained active in the safety-critical QP. The Crazyflie was then commanded to track a sequence of position references. The step values were chosen such that the nominal control input, $\boldsymbol{\mu}_{\mathrm{d}}$, would cause the MAV to exceed its flight limitations, necessitating intervention from the safety-critical QP.

To determine performance expectations prior to flight, we simulated the demonstration, and the results are depicted in Fig. 6(a)-(b). Then, with a baseline established, the flight profile was executed. The results from one example flight experiment are displayed in Fig. 6(c)-(d), and video of this flight is provided in the Supplementary Material.

Comparing the data, we see that the QP solver circuit performed nominally, and the resultant MAV flight closely matched the simulation. This highlights the functionality of the analog circuit and our hybrid safety-critical control architecture. Furthermore, from the data in Fig. 6(d), we can compute the accuracy of the analog QP solver versus the real-time digital QP solution (calculated using `quadprog`); the root-mean-square error of the analog solver is between 0.4 m/s^2 and 0.5 m/s^2 along all three inertial axes. This represents an error of 4 - 5% relative to the solver output range. These errors, produced by electrical crosstalk in the prototype analog circuit, resulted in minor deviations in the MAV flight path. We can reduce noise in the future with a more robust electrical design.

VI. CONCLUSIONS

We introduced a safety-critical control framework which leverages analog quadratic programming to drastically reduce computational cost and facilitate real-time embedded implementation, and we derived the procedure for synthesizing controllers using this framework. Additionally, we demonstrated the benefit of our hybrid computing architecture, in simulation, during an MAV safety-critical collision avoidance task. For this simulated flight, we illustrated the importance of high control frequencies when responding to a dynamically changing environment. We validated the functionality of our framework by executing a real-time flight control experiment and demonstrated the benefits of hybrid computing. Moving forward, we will expand our experimental results to include real-time resistor variability using memristors, enabling a broad range of safety-critical control applications. We are also working to fully-embedded our hybrid architecture onboard an MAV system for greater practical applicability.

ACKNOWLEDGMENT

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via 2021-21090200005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

REFERENCES

[1] H. M. James, N. B. Nichols, and R. S. Phillips, *Theory of Servomechanisms*. McGraw-Hill Book Company, Inc., 1947.

[2] S. Bennett, "History of automatic control to 1960: An overview," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 3008–3013, 1996.

[3] F. J. Mullin, "The analysis and compensation of nonlinear sampled-data feedback systems," Ph.D. dissertation, University of California, Berkeley, 1958.

[4] N. S. Nise, *Control Systems Engineering*. Wiley, 2019.

[5] H. K. Khalil, *Nonlinear Systems*. Prentice-Hall, Inc., 2002.

[6] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[7] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[8] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, "3d dynamic walking on stepping stones with control barrier functions," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 827–834.

[9] T. D. Son and Q. Nguyen, "Safety-critical control for non-affine nonlinear systems with application on autonomous vehicle," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 7623–7628.

[10] A. Manjunath and Q. Nguyen, "Safe and robust motion planning for dynamic robotics via control barrier functions," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 2122–2128.

[11] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[12] B. Xu and K. Sreenath, "Safe teleoperation of dynamic uavs through control barrier functions," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7848–7855.

[13] V. Freire and X. Xu, "Flatness-based quadcopter trajectory planning and tracking with continuous-time safety guarantees," *arXiv*, 2021.

[14] L. Wang, A. D. Ames, and M. Egerstedt, "Safe certificate-based maneuvers for teams of quadrotors using differential flatness," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3293–3298.

[15] R. S. Frackowiak, *Human Brain Function*. Elsevier, 2003.

[16] B. Chen, H. Yang, B. Song, D. Meng, X. Yan, Y. Li, Y. Wang, P. Hu, T.-H. Ou, M. Barnell, Q. Wu, H. Wang, and W. Wu, "A memristor-based hybrid analog-digital computing platform for mobile robotics," *Science Robotics*, vol. 5, no. 47, p. eabb6938, 2020.

[17] S. Vichik and F. Borrelli, "Solving linear and quadratic programs with an analog circuit," *Computers & Chemical Engineering*, vol. 70, pp. 160–171, 2014.

[18] S. Vichik, "Quadratic and linear optimization with analog circuits," Ph.D. dissertation, University of California, Berkeley, 2015.

[19] S. Vichik, M. Arcak, and F. Borrelli, "Stability of an analog optimization circuit for quadratic programming," *Systems & Control Letters*, vol. 88, pp. 68–74, 2016.

[20] K. Deems, "High speed analog circuit for solving optimization problems," Master's thesis, University of California, Berkeley, 2015.

[21] Y. Zhao, X. He, T. Huang, and Q. Han, "Analog circuits for solving a class of variational inequality problems," *Neurocomputing*, vol. 295, pp. 142–152, 2018.

[22] A. Gangopadhyay, O. Chatterjee, and S. Chakrabartty, "Continuous-time optimization using sub-threshold current-mode growth transform circuits," in *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2018, pp. 246–249.

[23] I. Matei, A. Feldman, and J. de Kleer, "Analog implementation of optimization algorithms: A distributed optimization view," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 7170–7175.

[24] T. Skibik and A. A. Adegbege, "An architecture for analog vlsi implementation of embedded model predictive control," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 4676–4681.

[25] M. Di Marco, M. Forti, L. Pancioni, G. Innocenti, and A. Tesi, "Memristor neural networks for linear and quadratic programming problems," *IEEE Transactions on Cybernetics*, vol. 52, no. 3, pp. 1822–1835, 2022.

[26] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

[27] Q. Nguyen and K. Sreenath, "Exponential control barrier functions for enforcing high relative-degree safety-critical constraints," in *2016 American Control Conference (ACC)*, 2016, pp. 322–328.

[28] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.

[29] F. A. Potra and S. J. Wright, "Interior-point methods," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1, pp. 281–302, 2000.

[30] A. Isidori, *Nonlinear Control Systems*. Springer, 1985.

[31] G. Wu and K. Sreenath, "Safety-critical control of a planar quadrotor," in *2016 American Control Conference (ACC)*, 2016, pp. 2252–2258.

[32] ——, "Safety-critical control of a 3d quadrotor with range-limited sensing," in *Dynamic Systems and Control Conference*, Oct 2016.

[33] R. M. Bena, X.-T. Nguyen, X. Yang, A. A. Calderon, Y. Chen, and N. O. Pérez-Arancibia, "A multiplatform position control scheme for flying robotic insects," *Journal of Intelligent & Robotic Systems*, vol. 105, no. 1, p. 19, 2022.