Co-Design of Anytime Computation and Robust Control

Yash Vardhan Pant, Houssam Abbas, Kartik Mohta, Truong X. Nghiem, Joseph Devietti, Rahul Mangharam

APPENDIX

In this appendix we give the detailed mathematical derivation of the results of Section III. The controller is designed using a Robust Model Predictive Control (RMPC) approach via constraint restriction [17], [18], and augments it by an adaptation to the error-delay curve of the estimator. In order to ensure robust safety and feasibility, the key idea of the RMPC approach is to tighten the constraint sets iteratively to account for possible effect of the disturbances. As time progresses, this "robustness margin" is used in the MPC optimization with the nominal dynamics, i.e., the original dynamics where the disturbances are either removed or replaced by nominal disturbances. Because only the nominal dynamics are used, the complexity of the optimization is the same as for the nominal problem.

Since the controller only has access to the estimated state $\hat{x}$, we need to rewrite the plant's dynamics with respect to $\hat{x}$. The error between x_k and $\hat{x}_k$ is $e_k = x_k - \hat{x}_k$. At time step $k+1$ we have

$$\begin{aligned}\hat{x}_{k+1} &= x_{k+1} - e_{k+1} \\ &= Ax_k + B_1(\delta_k)u_{k-1} + B_2(\delta[k])u_k + w_k - e_{k+1},\end{aligned}$$

then, by writing $x_k = \hat{x}_k + e_k$, we obtain the dynamics

$$\hat{x}_{k+1} = A\hat{x}_k + B_1(\delta[k])u_{k-1} + B_2(\delta[k])u_k + \hat{w}_k \tag{1}$$

where $\hat{w}_k = w_k + Ae_k - e_{k+1}$. The set of possible values of $\hat{w}_k$ depends on the estimation accuracy at steps k and $k+1$ and is denoted by $\widehat{\mathcal{W}}(\epsilon[k], \epsilon[k+1])$, i.e., $\widehat{\mathcal{W}}(\epsilon, \epsilon') := \{w + Ae - e' \mid w \in \mathcal{W}, e \in \mathcal{E}(\epsilon), e' \in \mathcal{E}(\epsilon')\}$. Note that $\widehat{\mathcal{W}}(\epsilon[k], \epsilon[k+1])$ is independent of the time step k. It can be computed as $\widehat{\mathcal{W}}(\epsilon, \epsilon') = \mathcal{W} \oplus A\mathcal{E}(\epsilon) \oplus (-\mathcal{E}(\epsilon'))$ where the symbol $\oplus$ denotes the Minkowski sum of two sets.

The dynamics in (1) has a non-standard form where it depends on both the current and the previous control inputs. However we can expand the state variable to store the previous control input as

$$\hat{z}_k = \begin{bmatrix} \hat{x}_k \\ u_{k-1} \end{bmatrix} \in \mathbb{R}^{n+m}$$

and rewrite the dynamics as, for all $k \geq 0$,

$$\hat{z}_{k+1} = \hat{A}(\delta_k)\hat{z}_k + \hat{B}(\delta_k)u_k + \hat{F}\hat{w}_k. \tag{2}$$

*This work was supported by STARnet a Semiconductor Research Corporation program sponsored by MARCO and DARPA, NSF MRI-0923518 and the US Department of Transportation University Transportation Center Program

The Departments of Electrical and Systems Engineering and Computer and Information Sciences, University of Pennsylvania, Philadelphia, U.S.A. {yashpant,habbas,kmohta,nghiem,rahulm}@seas.upenn.edu, devietti@cis.upenn.edu

Here, the system matrices are

$$\begin{aligned}\hat{A}(\delta_k) &= \begin{bmatrix} A & B_1(\delta_k) \\ \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \end{bmatrix}, \\ \hat{B}(\delta_k) &= \begin{bmatrix} B_2(\delta_k) \\ \mathbb{I}_m \end{bmatrix}, \quad \hat{F} = \begin{bmatrix} \mathbb{I}_n \\ \mathbf{0}_{m\times n} \end{bmatrix}.\end{aligned} \tag{3}$$

Let the actual expanded state be $z_k = \left[x_k^T, u_{k-1}^T\right]^T$. Because the expanded state consists of both the plant's state and the previous control input, the state constraint $x_k \in X$ and the control constraint $u_k \in U$ are equivalent to the joint constraint $z_k \in X \times U$. We can now describe the RAMPC algorithm for the dynamics in (2).

A. Tractable RAMPC Algorithm

Let $N \geq 1$ be the horizon length of the RMPC optimization. Because the system matrices in the state equation (2) depend nonlinearly on the variables δ_k, the RMPC optimization is generally a mixed-integer nonlinear program, which is very hard to solve. To simplify the RMPC optimization to make it tractable, we fix the estimation mode for the entire RMPC horizon.

Let $\mathbb{P}_{(\delta,\epsilon)}(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1})$ denote the RMPC optimization problem at step $k \geq 0$ where the current state estimate is $\hat{x}_k$, the current estimation mode is $(\delta_k, \epsilon_k) \in \Delta$, the previous control input is u_{k-1}, and the estimation mode for the entire horizon (after step k) is fixed at $(\delta, \epsilon) \in \Delta$. Since the system matrices become constant now, if the stage cost $\ell(\cdot)$ is linear or positive semidefinite quadratic, each optimization problem $\mathbb{P}_{(\delta,\epsilon)}(\hat{x}_\cdot, \delta_\cdot, \epsilon_\cdot, u_{\cdot-1})$ is tractable and can be solved efficiently as we will show later. The RAMPC algorithm with Anytime Estimation is stated in Alg. 1 .

B. RMPC Formulation

We formulate the RMPC optimization $\mathbb{P}_{(\delta,\epsilon)}(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1})$ with respect to the nominal dynamics, which is the original dynamics in Eq. (2) but the disturbances are either removed or replaced by nominal disturbances. To ensure robust feasibility and safety, the state constraint set is tightened after each step using a candidate stabilizing state feedback control, and a terminal constraint is derived. In this RMPC formulation, we extend the approach in [17], [18]. At time step k, given $(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1})$ and for a fixed (δ, ϵ), we solve the following optimization

$$J^*_{\delta,\epsilon}(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1}) = \min_{\boldsymbol{u},\boldsymbol{x}} \sum_{j=0}^{N} \ell(\overline{x}_{k+j|k}, u_{k+j|k}) \tag{4a}$$

subject to, $\forall j \in \{0, \ldots, N\}$

$$\overline{z}_{k+j+1|k} = \hat{A}(\delta_{k+j|k})\overline{z}_{k+j|k} + \hat{B}(\delta_{k+j|k})u_{k+j|k} \tag{4b}$$

$$(\delta_{k+j+1|k}, \epsilon_{k+j+1|k}) = (\delta, \epsilon)$$

$$(\delta_{k|k}, \epsilon_{k|k}) = (\delta_k, \epsilon_k) \tag{4c}$$

$$\overline{x}_{k+j|k} = \begin{bmatrix} \mathbb{I}_n & \mathbf{0}_{n\times m} \end{bmatrix} \overline{z}_{k+j|k} \tag{4d}$$

$$\overline{z}_{k|k} = \left[\hat{x}_k^T, u_{k-1}^T\right]^T \tag{4e}$$

$$\overline{z}_{k+j|k} \in \mathcal{Z}_j\left(\epsilon_k, \epsilon\right) \tag{4f}$$

$$\overline{z}_{k+N+1|k} \in \mathcal{Z}_f\left(\epsilon_k, \epsilon\right) \tag{4g}$$

in which $\overline{z}$ and $\overline{x}$ are the variables of the nominal dynamics. The constraints of the optimization are explained below.

- (4b) is the nominal dynamics.
- (4c) states that the estimation mode is fixed at (δ, ϵ) except for the first time step when it is (δ_k, ϵ_k).
- (4d) extracts the nominal state $\overline{x}$ of the plant from the nominal expanded state $\overline{z}$.
- (4e) initializes the nominal expanded state at time step k by stacking the current state estimate and the previous control input.
- (4f) tightens the admissible set of the nominal expanded states by a sequence of shrinking sets.
- (4g) constrains the terminal expanded state to the terminal constraint set $\mathcal{Z}_f$.

The state constraint $\mathcal{Z}_j$: The tightened state constraint sets $\mathcal{Z}_j\left(\epsilon_k, \epsilon\right)$ are parameterized with two parameters ϵ_k and ϵ. They are defined as follows, for all $j \in \{0, \ldots, N\}$

$$\mathcal{Z}_0(\epsilon_k, \epsilon) = \mathcal{Z} \ominus \hat{F}\mathcal{E}(\epsilon_k) \tag{5}$$

$$\mathcal{Z}_{j+1}(\epsilon_k, \epsilon) = \mathcal{Z}_j(\epsilon, \epsilon) \ominus L_j\hat{F}\widehat{\mathcal{W}}(\epsilon_k, \epsilon) \tag{6}$$

in which the symbol $\ominus$ denotes the Pontryagin difference between two sets. The set $\mathcal{Z}$ combines the constraints for both the plant's state and the control input: $\mathcal{Z} = X \times U$. The matrix L_j is the state transition matrix for the nominal dynamics in (4b) under a candidate state feedback gain $K_j(\delta)$, for $j \in \{0, \ldots, N\}$

$$L_0 = \mathbb{I} \tag{7}$$

$$L_{j+1} = (\hat{A}(\delta) + \hat{B}(\delta)K_j(\delta))L_j \tag{8}$$

Note that the possibly time-varying sequence $K_j(\delta)$ is designed for each choice of δ (i.e., the system matrices $\hat{A}(\delta)$ and $\hat{B}(\delta)$), hence L_j depends on δ; however we write L_j for brevity. The candidate control $K_j(\delta)$ is designed to stabilize the nominal system (4b), desirably as fast as possible so that the sets $\mathcal{Z}_j$ are shrunk as little as possible. In particular, if $K_j(\delta)$ renders the nominal system nilpotent after $M < N$ steps then $L_j = \mathbf{0}$ for all $j \geq M$, therefore $\mathcal{Z}_j\left(\epsilon_k, \epsilon\right) = \mathcal{Z}_M\left(\epsilon_k, \epsilon\right)$ for all $j > M$.

The terminal constraint $\mathcal{Z}_f$: $\mathcal{Z}_f$ is given by

$$\mathcal{Z}_f\left(\epsilon_k, \epsilon\right) = \mathcal{C}(\delta, \epsilon) \ominus L_N\hat{F}\widehat{\mathcal{W}}(\epsilon_k, \epsilon) \tag{9}$$

where $\mathcal{C}(\delta, \epsilon)$ is a robust control invariant admissible set for δ [**?**], i.e., there exists a feedback control law $u = \kappa(z)$ such that $\forall z \in \mathcal{C}(\delta, \epsilon)$ and $\forall w \in \widehat{\mathcal{W}}(\epsilon, \epsilon)$

$$\hat{A}(\delta)z + \hat{B}(\delta)\kappa(z) + L_N\hat{F}w \in \mathcal{C}(\delta, \epsilon) \tag{10}$$

$$z \in \mathcal{Z}_N\left(\epsilon, \epsilon\right) \tag{11}$$

We remark that $\mathcal{C}(\delta, \epsilon)$ does not depend on (δ_k, ϵ_k), therefore it can be computed offline for each mode (δ, ϵ).

C. Proofs of Feasibility

The RMPC formulation of the previous section, with a fixed estimation mode $(\delta, \epsilon) \in \Delta$, is designed to ensure that the control problem is robustly feasible, as stated in the following theorem.

Theorem 0.1 (Robust Feasibility of RAMPC): For any estimation mode (δ, ϵ), if $\mathbb{P}_{(\delta,\epsilon)}(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1})$ is feasible then the system (2)controlled by the RAMPC and subjected to disturbances constrained by $w_k \in \mathcal{W}$ robustly satisfies the state constraint $x_k \in X$ and the control input constraint $u_k \in U$, and all subsequent optimizations $\mathbb{P}_{\delta,\epsilon}(\hat{x}_k, \delta[k], \epsilon[k], u_{k-1})$, $\forall k > k_0$, are feasible.

Proof: We will prove the theorem by recursion. We will show that if at any time step k the RMPC problem $\mathbb{P}_{\delta,\epsilon}(\hat{x}_k, \delta[k], \epsilon[k], u_{k-1})$ is feasible and feasible control input $u_k = u^\star_{k|k}$ is applied with estimation mode $(\delta[k+1], \epsilon[k+1]) = (\delta, \epsilon)$ then u_k is admissible and at the next time step $k+1$, the actual plant's state x_{k+1} is inside X and the optimization $\mathbb{P}_{\delta,\epsilon}(\hat{x}_{k+1}, \delta[k+1], \epsilon[k+1], u_k)$ is feasible for all disturbances. Then we can conclude the theorem because, by recursion, feasibility at time step k_0 implies robust constraint satisfaction and feasibility at time step $k_0 + 1$, and so on at all subsequent time steps.

Suppose $\mathbb{P}_{\delta,\epsilon}(\hat{x}_k, \delta[k], \epsilon[k], u_{k-1})$ is feasible. Then it has a feasible solution $\left(\{\overline{z}^\star_{k+j|k}\}_{j=0}^{N+1}, \{u^\star_{k+j|k}\}_{j=0}^{N}\right)$ that satisfies all the constraints in (4). Now we will construct a feasible candidate solution for $\mathbb{P}_{\delta,\epsilon}(\hat{x}_{k+1}, \delta[k+1], \epsilon[k+1], u_k)$ at the next time step by shifting the above solution by one step. Consider the following candidate solution for $\mathbb{P}_{\delta,\epsilon}(\hat{x}_{k+1}, \delta[k+1], \epsilon[k+1], u_k)$:

$$\overline{z}_{k+j+1|k+1} = \overline{z}^\star_{k+j+1|k} + L_j\hat{F}\hat{w}_k \tag{12a}$$

$$\overline{z}_{k+N+2|k+1} = \hat{A}\left(\delta\right)\overline{z}_{k+N+1|k+1} + \hat{B}\left(\delta\right)u_{k+N+1|k+1} \tag{12b}$$

$$u_{k+i+1|k+1} = u^\star_{k+i+1|k} + K_i\left(\delta\right)L_i\hat{F}\hat{w}_k \tag{12c}$$

$$u_{k+N+1|k+1} = \kappa\left(\overline{z}_{k+N+1|k+1}\right) \tag{12d}$$

in which $j \in \{0, \ldots, N\}$, $i \in \{0, \ldots, N-1\}$, and $\kappa(\cdot)$ is the feedback control law for the invariant set $\mathcal{C}(\delta, \epsilon)$ that is used in the terminal set. We first show that the input and state constraints are satisfied for u_k and x_{k+1}, then we will prove the feasibility of the above candidate solution for $\mathbb{P}_{\delta,\epsilon}(\hat{x}_{k+1}, \delta[k+1], \epsilon[k+1], u_k)$.

Validity of the applied input and the next state: The next plant's state is

$$\begin{aligned}
x_{k+1} &= Ax_k + B_1(\delta[k])\,u_{k-1} + B_2(\delta[k])\,u_k + w_k \\
&= A(\hat{x}_k + e_k) + B_1(\delta[k])\,u_{k-1} + B_2(\delta[k])\,u^\star_{k|k} + w_k \\
&= \begin{bmatrix} A & B_1(\delta[k]) \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ u_{k-1} \end{bmatrix} + B_2(\delta[k])\,u^\star_{k|k} \\
&\qquad + e_{k+1} + (w_k + Ae_k - e_{k+1})
\end{aligned}$$

in which $e_{k+1} \in \mathcal{E}(\epsilon)$ and $(w_k + Ae_k - e_{k+1}) \in \widehat{\mathcal{W}}(\epsilon[k], \epsilon)$. Note that $\overline{z}^\star_{k|k} = \left[\hat{x}_k^T, u_{k-1}^T\right]^T$. Hence we have

$$\begin{aligned}
\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} &= \hat{A}(\delta[k])\overline{z}^\star_{k|k} + \hat{B}(\delta[k])u^\star_{k|k} \\
&\qquad + \hat{F}e_{k+1} + \hat{F}(w_k + Ae_k - e_{k+1}) \\
&= \overline{z}^\star_{k+1|k} + \hat{F}e_{k+1} + \hat{F}(w_k + Ae_k - e_{k+1})
\end{aligned}$$

where we use the dynamics in (4b). From (4f) and (6), $\overline{z}^\star_{k+1|k}$ satisfies $\overline{z}^\star_{k+1|k} \in \mathcal{Z}_1(\epsilon[k], \epsilon) = \mathcal{Z} \ominus \hat{F}\mathcal{E}(\epsilon) \ominus \hat{F}\widehat{\mathcal{W}}(\epsilon[k], \epsilon)$. It follows that $\left[x_{k+1}^T, u_k^T\right]^T \in \mathcal{Z} = X \times U$, therefore $x_{k+1} \in X$ and $u_k \in U$.

Initial condition: We have from (2) that $\hat{z}_{k+1} = \hat{A}(\delta[k])\hat{z}_k + \hat{B}(\delta[k])u_k + \hat{F}\hat{w}_k$. On the other hand, by (12a),

$$\begin{aligned}
\overline{z}_{k+1|k+1} &= \overline{z}^\star_{k+1|k} + L_0\hat{F}\hat{w}_k \\
&= \hat{A}(\delta[k])\overline{z}^\star_{k|k} + \hat{B}(\delta[k])u^\star_{k|k} + L_0\hat{F}\hat{w}_k.
\end{aligned}$$

Note that $\overline{z}^\star_{k|k} = \hat{z}_k$, $u_k = u^\star_{k|k}$, and $L_0 = \mathbb{I}$. Therefore $\overline{z}_{k+1|k+1} = \hat{z}_{k+1}$, hence the initial condition is satisfied.

Dynamics: We show that the candidate solution satisfies the dynamics constraint in Eq. (4b). For $0 \leq j < N$ we have

$$\begin{aligned}
&\overline{z}_{k+j+2|k+1} \\
&= \overline{z}^\star_{k+j+2|k} + L_{j+1}\hat{F}\hat{w}_k \\
&= \hat{A}(\delta)\,\overline{z}^\star_{k+j+1|k} + \hat{B}(\delta)u^\star_{k+j+1|k} + L_{j+1}\hat{F}\hat{w}_k \\
&= \hat{A}(\delta)\left(\overline{z}_{k+j+1|k+1} - L_j\hat{F}\hat{w}_k\right) \\
&\quad + \hat{B}(\delta)\left(u_{k+j+1|k+1} - K_j(\delta)\,L_j\hat{F}\hat{w}_k\right) + L_{j+1}\hat{F}\hat{w}_k \\
&= \hat{A}(\delta)\,\overline{z}_{k+j+1|k+1} + \hat{B}(\delta)u_{k+j+1|k+1} \\
&\quad - \left(\hat{A}(\delta) + \hat{B}(\delta)K_j(\delta)\right)L_j\hat{F}\hat{w}_k + L_{j+1}\hat{F}\hat{w}_k \\
&= \hat{A}(\delta)\,\overline{z}_{k+j+1|k+1} + \hat{B}(\delta)u_{k+j+1|k+1}
\end{aligned}$$

where the equality in (8) is used to derive the last equality. Therefore the dynamics constraint is satisfied for all $0 \leq j < N$. For $j = N$, the constraint is satisfied by construction by (12b).

State constraints: We need to show that $\overline{z}_{(k+1)+j|k+1} \in \mathcal{Z}_j(\epsilon, \epsilon)$ for all $j \in \{0, \ldots, N\}$. Consider any $0 \leq j < N$. (6) states that $\mathcal{Z}_{j+1}(\epsilon[k], \epsilon) = \mathcal{Z}_j(\epsilon, \epsilon) \ominus L_j\hat{F}\widehat{\mathcal{W}}(\epsilon[k], \epsilon)$. From the construction of the candidate solution we have $\overline{z}_{k+j+1|k+1} = \overline{z}^\star_{k+j+1|k} + L_j\hat{F}\hat{w}_k$, where $\hat{w}_k \in \widehat{\mathcal{W}}(\epsilon[k], \epsilon)$ and $\overline{z}^\star_{k+j+1|k} \in \mathcal{Z}_{j+1}(\epsilon[k], \epsilon)$. By definition of the Pontryagin difference, we conclude that $\overline{z}_{k+j+1|k+1} \in \mathcal{Z}_j(\epsilon, \epsilon)$ for all $j \in \{0, \ldots, N-1\}$.

At $j = N$ the candidate solution in (12a) gives us $\overline{z}_{(k+1)+N|k+1} = \overline{z}^\star_{k+N+1|k} + L_N\hat{F}\hat{w}_k$. Because $\overline{z}^\star_{k+N+1|k} \in \mathcal{Z}_f(\epsilon[k], \epsilon) = \mathcal{C}(\delta, \epsilon) \ominus L_N\hat{F}\widehat{\mathcal{W}}(\epsilon[k], \epsilon)$ and $\hat{w}_k \in \widehat{\mathcal{W}}(\epsilon[k], \epsilon)$, we have $\overline{z}_{(k+1)+N|k+1} \in \mathcal{C}(\delta, \epsilon)$. The definition of $\mathcal{C}(\delta, \epsilon)$ in (10) implies $\mathcal{C}(\delta, \epsilon) \subseteq \mathcal{Z}_N(\epsilon, \epsilon)$. Therefore $\overline{z}_{(k+1)+N|k+1} \in \mathcal{Z}_N(\epsilon, \epsilon)$.

Terminal constraint: We need to show that $\overline{z}_{k+N+2|k+1} \in \mathcal{Z}_f(\epsilon, \epsilon) = \mathcal{C}(\delta, \epsilon) \ominus L_N\hat{F}\widehat{\mathcal{W}}(\epsilon, \epsilon)$. Add $L_N\hat{F}\hat{w}$, for any $w \in \widehat{\mathcal{W}}(\epsilon, \epsilon)$, to both sides of (12b) and note that $u_{k+N+1|k+1} = \kappa\left(\overline{z}_{k+N+1|k+1}\right)$, we have

$$\begin{aligned}
\overline{z}_{k+N+2|k+1} + L_N\hat{F}\hat{w} &= \hat{A}(\delta)\,\overline{z}_{k+N+1|k+1} \\
&\qquad + \hat{B}(\delta)\,\kappa\left(\overline{z}_{k+N+1|k+1}\right) + L_N\hat{F}\hat{w}.
\end{aligned}$$

It follows from $\overline{z}_{k+N+1|k+1} \in \mathcal{C}(\delta, \epsilon)$ and from the definition of the invariant control invariant admissible set $\mathcal{C}(\delta, \epsilon)$ (Eq.(10)) that $\overline{z}_{k+N+2|k+1} + L_N\hat{F}\hat{w} \in \mathcal{C}(\delta, \epsilon)$ for all $w \in \widehat{\mathcal{W}}(\epsilon, \epsilon)$. Then by definition of the Pontryagin difference, we conclude that $\overline{z}_{k+N+2|k+1} \in \mathcal{C}(\delta, \epsilon) \ominus L_N\hat{F}\widehat{\mathcal{W}}(\epsilon, \epsilon) = \mathcal{Z}_f(\epsilon, \epsilon)$. ■

The control algorithm in Alg. 1 , in each time step k, solves $\mathbb{P}_{(\delta,\epsilon)}(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1})$ for each estimation mode $(\delta, \epsilon) \in \Delta$ and selects the control input u_k and the next estimation mode $(\delta_{k+1}, \epsilon_{k+1})$ corresponding to the best total cost $J_{(\delta,\epsilon)}$. Therefore, during the course of control, the algorithm may switch between the estimation modes in Δ depending on the system's state. Thm. 0.2 states that if the control algorithm Alg. 1 is feasible in its first time step then it will be robustly feasible and the state and control input constraints are also robustly satisfied.

Theorem 0.2: If at the initial time step there exists $(\delta, \epsilon) \in \Delta$ such that $\mathbb{P}_{(\delta,\epsilon)}(\hat{x}_0, \delta_0, \epsilon_0, u_{0-1})$ is feasible then the system Eq. 1 controlled by Alg. 1 and subjected to disturbances constrained s.t. $w_k \in \mathcal{W}, \forall k \geq 0$ robustly satisfies the state constraint $x_k \in X, \forall k \geq 0$ and the control input constraint $u_k \in U, \forall k \geq 0$, and all subsequent iterations of the algorithm are feasible.

Proof: The Theorem can be proved by recursively applying Thm. 0.1. Indeed, suppose at time step k the algorithm is feasible and results in control input u_k and next estimation mode $(\delta_{k+1}, \epsilon_{k+1})$, then $\mathbb{P}_{(\delta_{k+1},\epsilon_{k+1})}(\hat{x}_k, \delta_k, \epsilon_k, u_{k-1})$ is feasible. By Thm. 0.1, $u_k \in U$ and at the next time step $k+1$, $x_{k+1} \in X$ and $\mathbb{P}_{(\delta_{k+1},\epsilon_{k+1})}(\hat{x}_{k+1}, \delta_{k+1}, \epsilon_{k+1}, u_{k+1-1})$ is also feasible, hence the algorithm is feasible. Therefore, the Theorem holds by induction. ■

I. Experimental Setup

To evaluate our methodology on a real platform, we applied it to a hexrotor with the Odroid-U3 as a computation platform, running the Robot Operating System (ROS) [**?**] in Ubuntu. For the evaluation, the hexrotor is tasked with repeatedly following a given circular trajectory. As can be seen in Fig. 3, the visual odometry algorithm can occcasionaly take a long time to give a pose estimate. In our formulation we have assumed that the estimator satisfies the

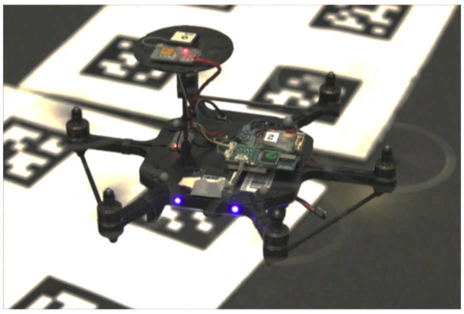

Fig. 1. Autonomous hexrotor with downward-facing camera flying over synthetic features.

(δ, ϵ) contract requested by the controller. Thus, to ensure that the estimator fulfills the contract and that the mathematical guarantees provided by our RAMPC formulation hold, instead of using the visual odometry algorithm to fly the robot, we injected delays and errors into the measurements from Vicon, which is a high accuracy localization system. These delays and errors were selected from the Δ curve obtained by profiling the SVO algorithm (see Section I-A). The hexrotor flies using these pose estimates and our RAMPC Algorithm for both the position control and setting the time deadline for the next estimate. The RAMPC has the positions and velocities in the 3-axes as its states (x), and generates control inputs in the form of desired thrust, roll and pitch (yaw is set to 0) in order to compute a given reference $x_{ref}(t)$ for a low-level controller to track. The RAMPC is coded in CVXGEN [**?**] and the generated C Code is integrated in the ROS module for position control of the hexrotor, running at 20Hz. The sets $\mathcal{Z}_j$ are done offline in MATLAB and then used in CVXGEN as Polyhedron type constraints. The constraint set X defines a safe set of positions and velocities in the flying area. The constraint set U of inputs keeps desired pitch and roll magnitudes less than 30 degrees and desired thrust within limits of the hex-rotor abilities. Later in this section we show that our approach dynamically schedules different modes of the contract-based perception and estimation algorithm at run-time and also controls the dynamical system in an energy-efficient way while providing good tracking performance. In the evaluation subsection we will compare our results to a baseline Model Predictive Control algorithm that does not leverage co-design and operates at a fixed (δ, ϵ) mode of the perception/estimation algorithm.

A. Profiling the perception and estimation pipeline

Recall that the control algorithm needs the profiled Δ curve for the estimator. In our experimental setup, the estimator is given by the SVO algorithm. Fig. 2 shows the bound on estimation error and the 90^{th} percentile execution times. This is obtained by varying the maximum number of corners knob in SVO, denoted by $\#C$, and flying extensively over a relatively feature-rich environment for each value of the knob (Fig. 1). The estimation error is computed using ground truth position obtained through the Vicon motion capture system. We profiled the performance for the same trajectory with different settings of the odometry offline by logging images and IMU data in-flight, and then running the visual odometry code on the Odroid-U3 offline. This accurately recreates the in-flight environment that is present for the visual odometry algorithm and this profiling is then used online for making in-flight decisions by the control algorithm.

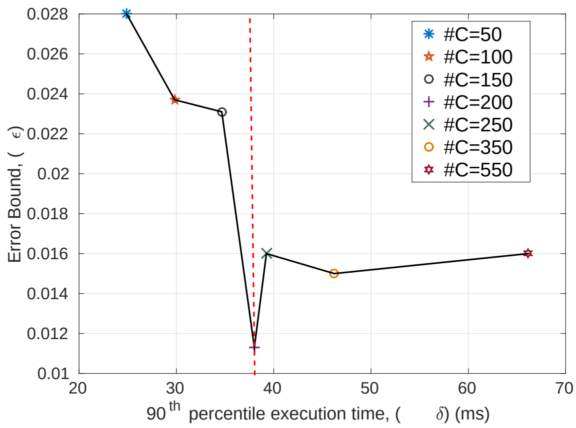

Fig. 2. Error-delay curve for the SVO algorithm running on the Odroid-U3 with different settings of maximum number of features ($\#C$) to detect and track. The vertical line shows the cut-off for maximum delay and the SVO settings that are allowable for closed loop control of a hexrotor at 20Hz.

Also needed for the control optimization is a measure of the power consumption by SVO at different values of the knob $\#C$. Power measurements are made using the Odroid Smart Power meter [**?**], which measures consumption at 10Hz to milliwatt precision. To avoid the physical challenges of fitting the power meter onto our hexrotor platform, we measure the power consumption of the Odroid board on the ground, running the same controller and vision workloads as it does during flight as explained above, and at different knob settings. We measure the power consumption of the entire Odroid board, including CPU and DRAM power consumption. Since the profiling of power is done offline with other peripherals plugged into the odroid (e.g. a monitor and keyboard), we measure the idle power of the Odroid and subtract that from the power measurements when the SVO algorithm is running on it in different modes. This gives us a more accurate measure of the workload that the visual odometry task is responsible for. This offline profiling now allows us to formulate the co-design problem for the hexrotor and experimentally evaluate our methods.

References

[1] M. Boddy and T. Dean, "Solving Time-dependent Planning Problems," *Joint Conf. on AI*, pp. 979–984, 1989.

[2] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI Magazine*, vol. 17, no. 3, 1996.

[3] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime Search in Dynamic Graphs," *Artif. Intell.*, vol. 172, no. 14, pp. 1613–1643, 2008.

[4] M. Wellman and C. L. Liu, "State-Space Abstraction for Anytime Evaluation of Probabilistic Networks," *Conf. on Uncertainty in AI*, 1994.

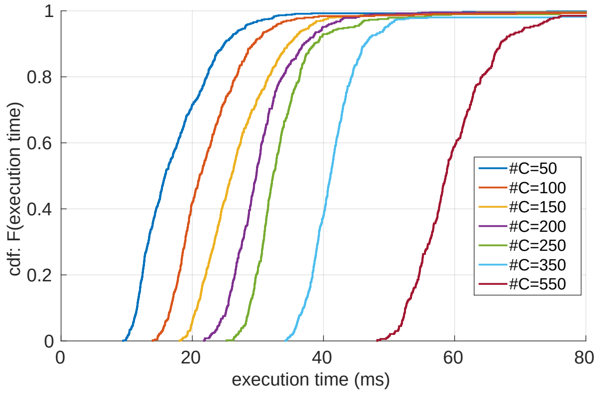

Fig. 3. Cumulative distribution of profiled execution times for visual odometry running on the Odroid-U3 for varying maximum number of corners from the SVO algorithm.

[5] R. Mangharam and A. Saba, "Anytime Algorithms for GPU Architectures," in *Proc. of the IEEE Real-Time Systems Symposium*, 2011.

[6] D. Quevedo and V. Gupta, "Sequence-based anytime control," *IEEE Trans. Autom. Control*, vol. 58, no. 2, pp. 377–390, Feb 2013.

[7] R. Bhattacharya and G. J. Balas, "Anytime control algorithm: Model reduction approach," *Journal of Guidance and Control*, vol. 27, no. 5, pp. 767–776, 2004.

[8] D. Fontanelli, L. Greco, and A. Bicchi, "Anytime control algorithms for embedded real-time systems," in *Hybrid Systems: Computation and Control*. Springer, 2008, pp. 158–171.

[9] G. Frehse, A. Hamann, S. Quinton, and M. Woehrle, "Formal analysis of timing effects on closed-loop properties of control software," in *Real-Time Systems Symposium (RTSS), 2014 IEEE*, Dec 2014, pp. 53–62.

[10] D. de Niz, L. Wrage, N. Storer, A. Rowe, and R. Rajukar, "On Resource Overbooking in an Unmanned Aerial Vehicle," *IEEE/ACM Third International Conference on Cyber-Physical Systems*, 2012.

[11] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek, "A Resource Allocation Model for QoS Mgmt." *IEEE RTSS*, 1997.

[12] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard, "Managing performance vs. accuracy trade-offs with loop perforation," in *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, ser. ESEC/FSE '11, 2011.

[13] M. Carbin, S. Misailovic, and M. C. Rinard, "Verifying quantitative reliability for programs that execute on unreliable hardware," in *Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages and Applications*, 2013.

[14] R. St. Amant, A. Yazdanbakhsh, J. Park, B. Thwaites, H. Esmaeilzadeh, A. Hassibi, L. Ceze, and D. Burger, "General-purpose code acceleration with limited-precision analog computation," in *Proc. of the 41st Annual International Symposium on Computer Architecuture*, ser. ISCA '14, 2014.

[15] Y. V. Pant, K. Mohta, H. Abbas, T. X. Nghiem, J. Devietti, and R. Mangharam, "Co-design of anytime computation and robust control (supplemental)," Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, Tech. Rep. UPenn-ESE-15-324, May 2015, http://repository.upenn.edu/mlab_papers.

[16] E. Camacho and C. Bordons, *Model predictive control*. Springer Verlag, 2004.

[17] A. Richards and J. How, "Robust model predictive control with imperfect information," in *American Control Conference*, 2005, pp. 268–273.

[18] L. Chisci, J. A. Rossiter, and G. Zappa, "Systems with persistent disturbances: predictive control with restricted constraints," *Automatica*, vol. 37, no. 7, pp. 1019–1028, 2001.

[19] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast Semi-Direct Monocular Visual Odometry," in *Robotics and Automation (ICRA), 2014 IEEE Intl. Conf. on*. IEEE, 2014.

[20] "ODROID-U3," http://odroid.com/, accessed: 2015-05-13.