# Target Tracking with Dynamic Convex Optimization

Alec Koppel⋆, Andrea Simonetto§, Aryan Mokhtari⋆, Geert Leus§, and Alejandro Ribeiro⋆
⋆Dept. of ESE, University of Pennsylvania, 200 South 33rd Street, Philadelphia, PA 19104, USA
§Dept. of EEMCS, Delft University of Technology, 2826 CD Delft, The Netherlands

***Abstract*—We develop a framework for trajectory tracking in dynamic settings, where an autonomous system is charged with the task of remaining close to an object of interest whose position varies continuously in time. We model this scenario as a convex optimization problem with a time-varying objective function and propose an adaptive discrete-time sampling prediction-correction scheme to find and track the solution trajectory while sampling the problem data at a constant rate of $1/h$. We propose approximate gradient trajectory (AGT) and approximate Newton trajectory tracking (ANT) as prediction-correction algorithms that (i) analyze the iso-residual dynamics of the optimality conditions in the prediction step, (ii) use gradient descent and Newton's method in the correction step, respectively, and (iii) approximate the partial derivative of the objective by a first-order backward derivative for the prediction step. We establish that the asymptotic error incurred by both proposed methods behaves as $O(h^2)$, and in some cases as $O(h^4)$, which outperforms the state-of-the-art error bound of $O(h)$ for correction-only methods in the gradient-correction step. The utility of the methods is demonstrated in an object tracking problem executed by an autonomous system.**

## I. Introduction

Trajectory tracking refers to a problem in which an autonomous system is charged with the task of remaining close to a reference trajectory which varies continuously in time. This problem setting has a rich history in control [1]–[5]. As in [6], we view this problem as a dynamic signal processing problem, where at point in time $t$ an autonomous system aims to solve for its position trajectory $\boldsymbol{x}(t) \in \mathbb{R}^n$ by solving

$$\min_{\boldsymbol{x}(t)} \; f(\boldsymbol{x}(t); t), \tag{1}$$

where $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is a strongly convex function informing the quality of the trajectory $\boldsymbol{x}(t)$. We propose sampling the problem at discrete times $t_k$ and recursively updating the control input to the system based upon remaining sufficiently close to the optimality conditions of (1). This perspective has much in common with finite horizon optimal control [7] and rolling horizon planning [8, Chapter 10.3.4].

Path planning strategies in intelligent systems have been recently approached using gradient schemes which seek a minimum-energy path [9]–[11], or more sophisticated schemes which decompose the feasible space based upon the geometry [12] of the particular task and implement sequential convex programming [13], or covariant gradient information [14]. Such methods may be viewed as variants of *running* methods [15] which only react on how the optimality conditions vary in time.

In contrast, we propose a running-tracking method by analyzing the iso-residual dynamics of the optimality conditions to predict the correct solution. We approximate the partial derivative of the objective in the prediction step by a first-order backward derivative. Then, we correct the predicted variable by descending through the gradient descent or Newton's method steps. Combining these steps we propose the approximate gradient trajectory tracking (AGT) method that uses gradient descent in the correction step, and the approximate Newton trajectory tracking (ANT) algorithm which corrects the predicted variable using Newton's method (Section II). Asymptotically the proposed algorithms track the optimal trajectory up to an error bound dependent on the sampling increment $h$ (Section III). We observe that the proposed methods work well in practice in a mobile robotic trajectory tracking problem (Section IV). All proofs are given in [16].

This research was supported in part by STW under the D2S2 project from the ASSYS program (project 10561) and in part by NSF CAREER CCF-0952867, and ONR N00014-12-1-0997.

## II. Algorithm Development

In this section we introduce a class of algorithms for solving optimization problem (1) using prediction and correction steps. To do so, we generate a sequence of near optimal control inputs $\{\boldsymbol{x}_k\}$ by considering both how the solution changes in time and how far our current update is from optimality at time $t_k$.

### A. Approximate prediction step

Consider the optimal solution $\boldsymbol{x}^*(t_0) = \operatorname{argmin} f(\boldsymbol{x}; t_0)$ of the initial objective $f(\boldsymbol{x}; t_0)$ and the case where we do not require the initial position of the agent to be optimal, i.e. $\boldsymbol{x}_0 \neq \boldsymbol{x}^*(t_0)$. Define residual error for the gradient of the initial variable $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_0; t_0) = \boldsymbol{r}(0)$. To improve the position estimation $\boldsymbol{x}$, we set up a prediction-correction scheme, similar to a Kalman filter [17], [18]. We first *predict* the solution changes, and then *correct* this prediction by using descent methods to push the predicted variable towards the instantaneous optimizer. To derive the prediction step, we reformulate the time-varying problem (1) in terms of its optimality conditions, i.e. the following nonlinear system of $n$ equations

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*(t); t) = \boldsymbol{0}, \tag{2}$$

for each $t$. The problems in (1) and (2) are equivalent since the family of objective functions $f(\boldsymbol{x}; t)$ are strongly convex with respect to $\boldsymbol{x}$ and only the optimal solutions of (1) satisfy (2). For simplicity we henceforth make the trajectory's dependence on time implicit: $\boldsymbol{x} = \boldsymbol{x}(t)$.

Consider an arbitrary vector $\boldsymbol{x} \in \mathbb{R}^n$ (for example the approximate solution $\boldsymbol{x}_k$). The objective function gradient $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}; t)$ computed at point $\boldsymbol{x}$ is

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}; t) = \boldsymbol{r}(t) \tag{3}$$

where $\boldsymbol{r}(t) \in \mathbb{R}^n$ is the residual error. The prediction step aims to keep the residual error close to constant despite changes in the optimization problem, which is tantamount to predicting $\boldsymbol{x}_k$ to stay close to the iso-residual manifold. Define $\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}; t)$ as the partial Hessian of the objective $f(\boldsymbol{x}; t)$ with respect to $\boldsymbol{x}$, and $\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t)$ as the mixed partial derivative of the objective $f(\boldsymbol{x}; t)$. We aim to maintain the evolution of the trajectory close to the residual vector $\boldsymbol{r}(t)$, i.e.

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}; t) + \nabla_{\boldsymbol{xx}} f(\boldsymbol{x}; t)\delta\boldsymbol{x} + \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t)\delta t = \boldsymbol{r}(t), \tag{4}$$

**Algorithm 1** Approximate Gradient Tracking (AGT)

**Require:** Initial variable $\boldsymbol{x}_0$. Initial objective function $f(\boldsymbol{x}; t_0)$

**for** $k = 0, 1, 2, \dots$ **do**

  Predict the solution using the prior information [cf. (7)]

$$\boldsymbol{x}_{k+1|k} = \boldsymbol{x}_k - [\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)]^{-1} \tilde{\nabla}_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)\, h$$

  Acquire the updated function $f(\boldsymbol{x}; t_{k+1})$

  Initialize the sequence of corrected variables $\hat{\boldsymbol{x}}^0_{k+1} = \boldsymbol{x}_{k+1|k}$

  **for** $s = 0 : \tau - 1$ **do**

    Correct the variable by the projected gradient step [cf. (9)]

$$\hat{\boldsymbol{x}}^{s+1}_{k+1} = P_X\left[\hat{\boldsymbol{x}}^s_{k+1} - \gamma \nabla_{\boldsymbol{x}} f(\hat{\boldsymbol{x}}^s_{k+1}; t_{k+1})\right]$$

  **end for**

  Set the corrected variable $\boldsymbol{x}_{k+1} = \hat{\boldsymbol{x}}^\tau_{k+1}$

**end for**

**Algorithm 2** Approximate Newton Tracking (ANT)

**Require:** Initial variable $\boldsymbol{x}_0$. Initial objective function $f(\boldsymbol{x}; t_0)$

**for** $k = 0, 1, 2, \dots$ **do**

  Predict the solution using the prior information [cf. (7)]

$$\boldsymbol{x}_{k+1|k} = \boldsymbol{x}_k - [\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)]^{-1} \tilde{\nabla}_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)\, h$$

  Acquire the updated function $f(\boldsymbol{x}; t_{k+1})$

  Initialize the sequence of corrected variables $\hat{\boldsymbol{x}}^0_{k+1} = \boldsymbol{x}_{k+1|k}$

  **for** $s = 0 : \tau - 1$ **do**

    Correct the variable by the projected Newton step [cf. (10)]

$$\hat{\boldsymbol{x}}^{s+1}_{k+1} = P_X\left[\hat{\boldsymbol{x}}^s_{k+1} - \gamma \nabla_{\boldsymbol{xx}} f(\hat{\boldsymbol{x}}^s_{k+1}; t_{k+1})^{-1} \nabla_{\boldsymbol{x}} f(\hat{\boldsymbol{x}}^s_{k+1}; t_{k+1})\right]$$

  **end for**

  Set the corrected variable $\boldsymbol{x}_{k+1} = \hat{\boldsymbol{x}}^\tau_{k+1}$

**end for**

where $\delta \boldsymbol{x}$ and $\delta t$ are small perturbations of the position variable $\boldsymbol{x}$ and the time variable $t$, respectively. By subtracting (3) from (4) and dividing the resulting equation by the time variation $\delta t$, we obtain the continuous dynamical system

$$\dot{\boldsymbol{x}} = -[\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}; t)]^{-1} \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t), \tag{5}$$

where $\dot{\boldsymbol{x}} = \delta \boldsymbol{x} / \delta t$. We consider the discrete time approximation of (5), which amounts to sampling the problem at times $t_k$, for $k = 0, 1, 2, \dots$ . The prediction step consists of a discrete-time approximation of integrating (5) by using an Euler scheme. Let $\boldsymbol{x}_{k+1|k}$ be the predicted position variable based on the available information up to time $t$, then we may write the Euler integral approximation of (5) as

$$\boldsymbol{x}_{k+1|k} = \boldsymbol{x}_k - [\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)]^{-1} \nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)\, h. \tag{6}$$

Observe that the prediction step in (6) is computed by only incorporating information available at time $t$; however, the variable $\boldsymbol{x}_{k+1|k}$ is supposed to be close to the iso-residual manifold of the objective function at time $t_{k+1}$.

Algorithms that assume the variation of the objective in time is accessible were derived in [19]; however, frequently in applications this time variation is not known. For example, if the reference path the agent aims to follow is varying erratically, computing its velocity is difficult, and may be approximated instead. Consider the mixed partial derivative at time $t_k$ using the gradient of the objective with respect to $\boldsymbol{x}$ at times $t_k$ and $t_{k-1}$, that is, define the approximate partial mixed gradient $\tilde{\nabla}_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)$ as

$$\tilde{\nabla}_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) = \frac{1}{h}(\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k; t_k) - \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_{k-1}; t_{k-1})). \tag{7}$$

which is called a *first-order backward derivative* since it requires information of the first previous step for approximating the current mixed partial derivative. Substituting the partial mixed gradient $\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)$ in (6) by its approximation $\tilde{\nabla}_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)$ in (7) leads to the *approximate prediction step*

$$\boldsymbol{x}_{k+1|k} = \boldsymbol{x}_k - [\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)]^{-1} \tilde{\nabla}_{t\boldsymbol{x}} f(\boldsymbol{x}_k; t_k)\, h. \tag{8}$$

The predicted variable $\boldsymbol{x}_{k+1|k}$ is an initial estimate for the optimal solution of the objective function $f(\boldsymbol{x}; t_{k+1})$. This estimation can be corrected by descending through the optimal argument of the objective function $f(\boldsymbol{x}; t_{k+1})$.

### *B. Correction step*

To correct the predicted control input $\boldsymbol{x}_{k+1|k}$ we modify it towards the optimal argument of the objective at time $t_{k+1}$. We introduce the approximate gradient trajectory (AGT) algorithm that corrects the predicted variable $\boldsymbol{x}_{k+1|k}$ by executing projected gradient descent steps for the objective function $f(\boldsymbol{x}; t_{k+1})$. Notice that the number of steps that can be afforded depends on the sampling increment $h = t_{k+1} - t_k$. Define $\tau$ as the number of projected gradient descent steps used for correcting the predicted variable $\boldsymbol{x}_{k+1|k}$ and $\hat{\boldsymbol{x}}^s_{k+1}$ as the corrected position variable after executing $s$ gradient descent steps. The sequence of variables $\hat{\boldsymbol{x}}^s_{k+1}$ is initialized by $\hat{\boldsymbol{x}}^0_{k+1} = \boldsymbol{x}_{k+1|k}$ and updated as

$$\hat{\boldsymbol{x}}^{s+1}_{k+1} = P_X[\hat{\boldsymbol{x}}^s_{k+1} - \gamma \nabla_{\boldsymbol{x}} f(\hat{\boldsymbol{x}}^s_{k+1}; t_{k+1})], \tag{9}$$

where $P_X$ denotes the Euclidean projection operator onto the set $X$ and $\gamma > 0$ is the stepsize. After executing $\tau$ steps of (9) the decision variable $\boldsymbol{x}(t_{k+1}) := \boldsymbol{x}_{k+1} = \hat{\boldsymbol{x}}^\tau_{k+1}$ is computed.

Notice that AGT uses the steps of projected gradient descent for correcting the predicted control input $\boldsymbol{x}_{k+1|k}$. This process can be accelerated by implementing Newton's method that incorporates the second-order information of the objective in (1). We define the approximate Newton trajectory (ANT) algorithm which corrects the predicted variable $\boldsymbol{x}_{k+1|k}$ in (8) by the projected Newton's method. The prediction step of the ANT algorithm is identical to that of the AGT method [cf. (8)]; however, the correction step of ANT updates the predicted solution trajectory by applying $\tau$ steps of the Newton method. In particular, the predicted variable $\boldsymbol{x}_{k+1|k}$ is used for initializing the sequence of corrected variables $\hat{\boldsymbol{x}}^s_{k+1}$, i.e., $\hat{\boldsymbol{x}}^0_{k+1} := \boldsymbol{x}_{k+1|k}$. The sequence of corrected variables $\hat{\boldsymbol{x}}^s_{k+1}$ is updated using Newton steps as

$$\hat{\boldsymbol{x}}^{s+1}_{k+1} = P_X\left[\hat{\boldsymbol{x}}^s_{k+1} - \gamma \nabla_{\boldsymbol{xx}} f(\hat{\boldsymbol{x}}^s_{k+1}; t_{k+1})^{-1} \nabla_{\boldsymbol{x}} f(\hat{\boldsymbol{x}}^s_{k+1}; t_{k+1})\right]. \tag{10}$$

The control input at step $t_{k+1}$ for the ANT method is the outcome of $\tau$ iterations of (10), i.e., $\boldsymbol{x}_{k+1} = \hat{\boldsymbol{x}}^\tau_{k+1}$.

*Remark 1:* Notice that the prediction step of both AGT and ANT introduced in (6) requires computation of the partial Hessian inverse $[\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}_k; t_k)]^{-1}$, which has a computational complexity of order $O(n^3)$. This computation justifies the use of Newton's method for the correction step in ANT. On other hand, the number of projected gradient descent steps $\tau$ that AGT can execute for correcting the predicted variable is larger than the number of Newton's steps that ANT can accomplish in the same amount of time. We study the effect of the computational complexity difference of these algorithms in Section IV.

## III. Convergence Analysis

We turn to establishing that the prediction-correction schemes derived in Section II solve the continuous-time problem stated in (1) up to an error term which is dependent on the discrete-time

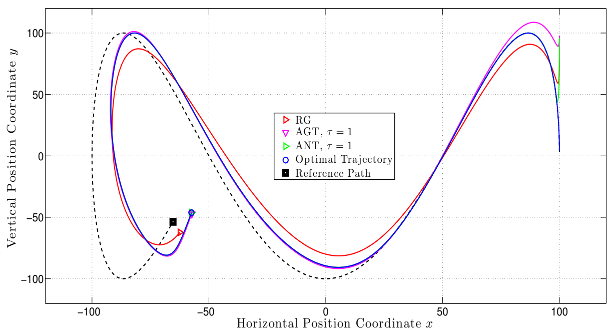


Fig. 1. Sample trajectories of the object to be tracked (dashed) and the trajectories generated by the different algorithms (continuous). All algorithms track the optimum effectively, yet AGT and ANT track $\boldsymbol{x}^*(t)$ closer than RG. Note that exact path-following is not the objective, as the robot aims to remain close to base $\boldsymbol{b} = [100, 100]$, yielding unexpected behavior of ANT near time $t = 300$.

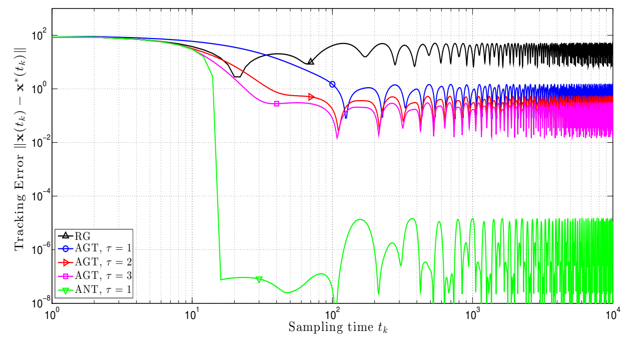


Fig. 2. Error $\|\boldsymbol{x}(t_k) - \boldsymbol{x}^*(t_k)\|$ versus sampling time $t_k$ for $h = 1$. Observe that AGT outperforms RG but achieves comparable performance across differing numbers of correction steps $\tau$. Moreover, ANT experiences superior error bounds relative to the first-order methods.

sampling interval. All proofs may be found in [16]. To do so, some technical conditions are required which we state below.

(A1) The solution trajectory $\boldsymbol{x}^*(t)$ of (1) is contained in the interior of a convex set $X \subseteq \mathbb{R}^n$ for each $t$.

(A2) The function $f(\boldsymbol{x}; t)$ is twice differentiable and $m$-strongly convex in $\boldsymbol{x} \in X$ and uniformly in $t$. I.e.,

$$\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}; t) \geq m\mathbf{I}, \quad \forall\ \boldsymbol{x} \in X, t. \tag{11}$$

(A3) The function $f(\boldsymbol{x}; t)$ has bounded second and third derivatives w.r.t. $\boldsymbol{x} \in X$ and $t$, $\forall \boldsymbol{x} \in X$ and $\forall t$, with constants

$$\|\nabla_{\boldsymbol{xx}} f(\boldsymbol{x}; t)\| \leqslant L, \|\nabla_{t\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant C_0, \|\nabla_{\boldsymbol{xxx}} f(x; t)\| \leqslant C_1$$
$$\|\nabla_{\boldsymbol{x}t\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant C_2, \quad \|\nabla_{tt\boldsymbol{x}} f(\boldsymbol{x}; t)\| \leqslant C_3. \tag{12}$$

Assumption (A1) is satisfied for $X = \mathbb{R}^n$, implying existence of a solution for (1) at each time $t$. Assumption (A2) ensures that Problem (1) has a *unique* solution for each time instance and that the Hessian of the objective $f(\boldsymbol{x}; t)$ is invertible, a commonly required condition in time-varying settings [20]–[23]. Assumption (A3) gives to the time-varying problem the boundedness required to ensure solution tracking. With unbounded derivatives, little may be said and tracking becomes intractable.

Under Assumptions (A1) - (A3), one may establish that the solution *mapping* $t \mapsto \boldsymbol{x}^*(t)$ is one-to-one and does not vary arbitrarily in time, as presented in [24, Theorem 2F.10], which allows AGT and ANT to converge to a neighborhood of the optimal solution. We present the convergence properties of the AGT method for different step-sizes in the following theorem.

*Theorem 1:* Denote the approximate gradient tracking algorithm generated by Algorithm 1 as $\{\boldsymbol{x}_k\}$. Let Assumptions (A1) - (A3) hold and define the constants $\rho$ and $\sigma$ as $\rho = (1 + \gamma^2 L^2 - \gamma m)^{1/2}$ and $\sigma = 1 + h(C_0 C_1/m^2 + C_2/m)$.

i) For any sampling increment $h$, if the stepsize satisfies $\gamma < m/L^2$, the sequence $\{\boldsymbol{x}_k\}$ converges to $\boldsymbol{x}^*(t_k)$ Q-linearly up to a bounded error as

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| \leqslant \rho^{\tau k} \|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\| + O(h) \tag{13}$$

ii) If the sampling increment $h$ and the step-size $\gamma > 0$ are chosen such $\rho^\tau \sigma < 1$, then the sequence $\{\boldsymbol{x}_k\}$ converges to $\boldsymbol{x}^*(t_k)$ Q-linearly up to a bounded error as,

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| \leqslant (\rho^\tau \sigma)^k \|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\| + O(h^2) \tag{14}$$

Theorem 1 states the convergence properties of the AGT algorithm for different choices of the parameters. In both cases, the linear convergence to a neighborhood is shown with convergence accuracy depending on the sampling increment $h$, the step-size $\gamma$, and the number of projected gradient descent steps $\tau$. Moreover, for particular step-size selections depending on smoothness properties of the objective, the asymptotic error bound converges up to either $O(h)$ or $O(h^2)$. The proof may be found in [16].

AGT uses only first-order information of the objective $f(\boldsymbol{x}; t_{k+1})$ to correct the predicted variable $\boldsymbol{x}_{k+1|k}$, while ANT uses Newton's method in the correction step. Similar to the advantages of Newton's method relative to gradient descent, we achieve faster convergence and more accurate estimation for ANT relative to AGT in the following theorem.

*Theorem 2:* Let $\{\boldsymbol{x}_k\}$ be the Approximate Newton Tracking algorithm generated by Algorithm 2, define $\sigma$ as in Theorem 1, and suppose Assumptions (A1) - (A3) hold. Define the cumulative approximation error

$$D = h^2 \left[ \frac{C_0 C_2}{2m^2} + \frac{C_3}{m} \right]. \tag{15}$$

Assume that the initial optimality gap $\|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\|$ can be written as $\|\boldsymbol{x}_0 - \boldsymbol{x}^*(t_0)\| = \beta D$ where $\beta > 0$ is a constant. Further, assume that the sampling increment $h$ is small enough that the upper bound for the cumulative approximation error $D$ is bounded above by 1.

i) If the condition $C_1 D/2m \leqslant \beta/(\sigma\beta + 1)^2$ is satisfied, then the sequence $\{\boldsymbol{x}_k\}$ generated by ANT converges as

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| \leqslant \beta D = O(h^2); \tag{16}$$

ii) Further, if $C_1/2m \leqslant \beta/(\sigma\beta + 1)^2$, then $\{\boldsymbol{x}_k\}$ satisfies

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*(t_k)\| \leqslant \beta D^2 = O(h^4). \tag{17}$$

Theorem 2 establishes that ANT tracks the optimal trajectory $\boldsymbol{x}^*(t_k)$ up to an error bound of order $O(h^2)$, which is comparable to the convergence result of AGT in (14). Moreover, when $C_1/2m \leqslant \beta/(\sigma\beta + 1)^2$, ANT achieves an error bound of order $O(h^4)$, as a result of the quadratic phase of Newton's method. See [16] for the proof.

## IV. Path-Following Experiments

As outlined in Section I, our application of interest is an autonomous system which is charged with the task of following an object whose position is varying continuously in time. Denote the reference trajectory of this object as a curve $\boldsymbol{y}(t)$, i.e. a function $\boldsymbol{y} : \mathbb{R}_+ \to \mathbb{R}^n$ and $\boldsymbol{x} \in \mathbb{R}^n$ be the decision variable of the robot,

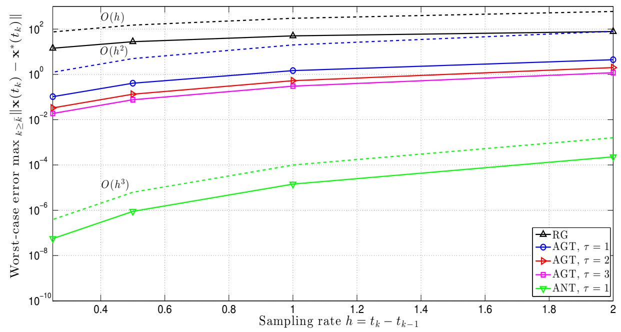


Fig. 3. Worst-case Error $\max_{k \geqslant \bar{k}} \|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|$ over sampling increment $h$ for $\bar{k} = 8 \times 10^3$. RG experiences a worst-case error comparable to $O(h)$, whereas AGT and ANT achieve $O(h^2)$ and $O(h^3)$, respectively, outperforming existing methods.

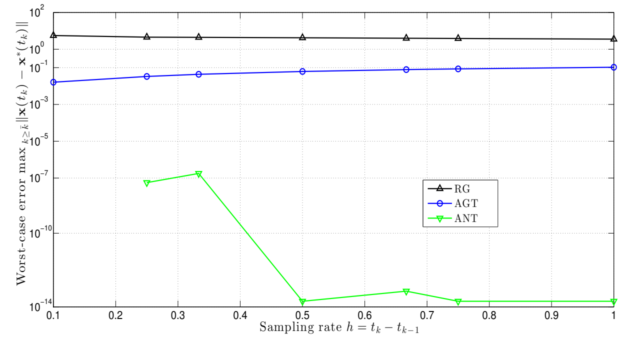


Fig. 4. Worst case Error $\max_{k \geqslant \bar{k}} \|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|$ versus sampling increment $h$ with fixed control loop latency. The number of correction steps $\tau$ is dynamically adjusted based on $h$. For $h < 1/4$, ANT is infeasible, and in this range AGT outperforms RG. Moreover, as soon as ANT may be afforded, it achieves superior performance.

in terms of the waypoint it aims to reach next. We aim to solve tracking problems of the form

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}; t) := \|\boldsymbol{x} - \boldsymbol{y}(t)\|^2 + \mu_1 \exp\left(\mu_2 \|\boldsymbol{x}_i - \boldsymbol{b}\|^2\right), \quad (18)$$

The optimal solution $\boldsymbol{x}^*(t)$ tracks the reference path $\boldsymbol{y}(t)$ while remaining close enough to the base $\boldsymbol{b}$, which may correspond to a recharging station or a domain constraint associated with maintaining viable communications.

For the numerical study considered here, we consider a planar example ($n = 2$) and fix $\mu_1 = 10^3$, $\mu_2 = 5 \times 10^{-3}$ with the base located at $\boldsymbol{b} = [100; 100]$. In addition, we suppose the target trajectory $\boldsymbol{y}(t)$ follows the specified path $\boldsymbol{y}(t) = 100[\cos(\omega t), \sin(3\omega t)]$ where $\omega = 0.01$. Moreover, the position domain is given as $X = [-150, 150] \times [-150, 150]$. We select step-size $\gamma = 0.05$. With these parameters, the target moves with maximum speed 3.16. This is comparable with the speed of current quad-rotors (max speed ∼10 m/s). In any practical setting, the actuation capability of an autonomous system is limited, i.e. we may only achieve some maximum velocity denoted as $v_{\max}$. The choice made for our numerical experiments is $v_{\max} = 4$, and thus we threshold the outputs of Algorithms 1 and 2 to account for this constraint.

We show the result of this experiment in terms of the actual reference path and trajectories generated by AGT and ANT in Figure 1 over a truncated time interval $0 < t < 300$ for $h = 1$. The reference trajectory $\boldsymbol{y}(t)$ is the dotted line, and the optimal continuous-time trajectory $\boldsymbol{x}^*(t)$ associated with solving (18) is in blue. By running gradient we mean a method which has no prediction step, and operates only by correction. Observe that the trajectories generated by the running gradient (RG), AGT, and ANT successfully track the optimal trajectory $\boldsymbol{x}^*(t)$, and consequently the reference path $\boldsymbol{y}(t)$ up to a small error.

This trend may be more easily observed in Figure 2 which shows the magnitude of the difference between the generated path and the optimal path $\|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|$. The approximate steady state errors achieved by RG, AGT, and ANT are respectively 10, $10^{-1}$, and $10^{-6}$. AGT experiences comparable levels of error across different values of $\tau$, the number of correction steps, and ANT far outperforms the other methods. This pattern is corroborated in Figure 3, which plots the worst-case optimality gap $\max_{k \geqslant \bar{k}} \|\boldsymbol{x}^*(t_k) - \boldsymbol{x}_k\|$ versus the sampling interval size $h$ for $\bar{k} = 8 \times 10^3$. In particular, we observe that RG experiences an error comparable to $O(h)$, as it theoretically guarantees, whereas our proposed methods AGT and ANT achieve a worst-case error of approximately $O(h^2)$ and $O(h^3)$, respectively. Moreover, when the problem (1) is sampled less often, i.e., when $h$ increases, the optimality gap increases.

Observe that ANT far outperforms the other methods; however, this performance gap ignores the increased computational cost associated with Newton steps. To obtain a more fair comparison, we consider how the different algorithms perform when the computation time to complete providing a control input is fixed. In this simulation setting, the most demanding task is the evaluation of the gradient and the Hessian, while the actual prediction or correction step is less critical (less than $1/10$ times). In particular, evaluating the Hessian requires *twice* the computational effort of evaluating the gradient, so a Newton step is three times slower than a gradient step. Note that since the prediction step is performed before a new target position is acquired, it may not be considered in the time latency for generating the next waypoint. Hence we only consider the delay associated with the correction step in this numerical experiment.

TABLE I
NUMBER OF CORRECTION STEPS $\tau$ TO KEEP THE SAME FEEDBACK LATENCY

| Sampling time $h$ [s] | 1/10 | 1/4 | 1/3 | 1/2 | 2/3 | 3/4 | 1 |
|---|---|---|---|---|---|---|---|
| RG | 1 | 3 | 4 | 6 | 8 | 9 | 12 |
| AGT | 1 | 3 | 4 | 6 | 8 | 9 | 12 |
| ANT | – | 1 | 1 | 2 | 2 | 3 | 4 |

We now outline how Table 1 is generated. We set at $\Delta t = h/10$ the allowable computational time for the correction step, and we set the gradient evaluation to require $1/120$ s (thus, e.g., the robot can perform only $\tau = 1$ gradient correction step for a sampling time of $h = 0.1$ s). With this as our basic unit of measurement, we fill in Table 1 with how many gradient evaluations $\tau$ may be afforded with increasing the sampling interval $h$. As previously noted, ANT requires three times the computation time of AGT, and consequently experiences too much latency to be used when $h = 0.1$ s. Notice that the difference between the computational costs of AGT and ANT becomes larger by increasing $n$.

For fixed computation time, i.e., control loop latency, the number of correction steps $\tau$ must be dynamically adjusted based on the sampling increment $h$. We repeat the path-following experiment in (18) for different algorithms when the computation time is fixed in Figure 4. In particular, considering the worst-case optimality gap versus sampling increment $h$, we see that if ANT may be afforded (for large $h$), it is much preferable to AGT. However, for small sampling intervals $h$, i.e., when one requires very low latencies in the control loop, ANT is infeasible.

# REFERENCES


[1] C. Castillo, W. Moreno, and K. Valavanis, "Unmanned helicopter waypoint trajectory tracking using model predictive control," in *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pp. 1–8, IEEE, 2007.

[2] A. Vahidi and A. Eskandarian, "Research advances in intelligent collision avoidance and adaptive cruise control," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 4, no. 3, pp. 143–153, 2003.

[3] T. Ardeshiri, M. Norrlöf, J. Löfberg, and A. Hansson, "Convex optimization approach for time-optimal path tracking of robots with speed dependent constraints," 2010.

[4] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *Automatic Control, IEEE Transactions on*, vol. 54, no. 10, pp. 2318–2327, 2009.

[5] M. G. Rabbat and R. D. Nowak, "Decentralized source localization and tracking," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 3, pp. iii–921, IEEE, 2004.

[6] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "D4L: Decentralized Dynamic Discriminative Dictionary Learning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (submitted)*, October 2015.

[7] R. M. Murray, "Optimization-based control," *California Institute of Technology, CA*, 2009.

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

[9] R. Abdolee, B. Champagne, and A. H. Sayed, "Estimation of space-time varying parameters using a diffusion LMS algorithm," *IEEE Transactions on Signal Processing*, vol. 62, no. 2, pp. 403–418, 2014.

[10] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pp. 802–807, IEEE, 1993.

[11] M. Rupp and A. H. Sayed, "A time-domain feedback analysis of filtered-error adaptive gradient algorithms," *IEEE Transactions on Signal Processing*, vol. 44, no. 6, pp. 1428–1439, 1996.

[12] D. Koditschek, "Robot planning and control via potential functions," pp. 349–367, 1989.

[13] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization.," in *Robotics: Science and Systems*, vol. 9, pp. 1–10, Citeseer, 2013.

[14] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 489–494, IEEE, 2009.

[15] D. Osmankovica and J. Velagic, "Gradient based adaptive trajectory tracking control for mobile robots," in *Information, Communication and Automation Technologies (ICAT), 2013 XXIV International Symposium on*, pp. 1–6, IEEE, 2013.

[16] A. Simonetto, A. Koppel, A. Mokhtari, G. Leus, and A. Ribeiro, "A Class of Prediction-Correction Methods for Time-Varying Convex Optimization," *In preparation*, 2015. Available at http://ens.ewi.tudelft.nl/~asimonetto/TimeVarying_part1.pdf.

[17] A. Ribeiro, G. B. Giannakis, and S. I. Roumeliotis, "SOI-KF: Distributed Kalman filtering with low-cost communications using the sign of innovations," *Signal Processing, IEEE Transactions on*, vol. 54, no. 12, pp. 4782–4795, 2006.

[18] U. A. Khan and J. M. Moura, "Distributing the Kalman filter for large-scale systems," *Signal Processing, IEEE Transactions on*, vol. 56, no. 10, pp. 4919–4935, 2008.

[19] A. Simonetto, A. Koppel, A. Mokhtari, G. Leeus, and A. Ribeiro, "Prediction-Correction Methods for Time-Varying Convex Optimization," in *Proceedings of the Asilomar Conference on Signals, Systems, and Computers (submitted)*, (Pacific Grove, USA), November 2015.

[20] A. Y. Popkov, "Gradient Methods for Nonstationary Unconstrained Optimization Problems," *Automation and Remote Control*, vol. 66, no. 6, pp. 883 – 891, 2005. Translated from Avtomatika i Telemekhanika, No. 6, 2005, pp. 38 – 46.

[21] A. L. Dontchev, M. I. Krastanov, R. T. Rockafellar, and V. M. Veliov, "An Euler-Newton Continuation method for Tracking Solution Trajectories of Parametric Variational Inequalities," *SIAM Journal of Control and Optimization*, vol. 51, no. 51, pp. 1823 – 1840, 2013.

[22] F. Y. Jakubiec and A. Ribeiro, "D-MAP: Distributed Maximum a Posteriori Probability Estimation of Dynamic Systems," *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 450 – 466, 2013.

[23] Q. Ling and A. Ribeiro, "Decentralized Dynamic Optimization Through the Alternating Direction Method of Multipliers," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1185 – 1197, 2014.

[24] A. L. Dontchev and R. T. Rockafellar, *Implicit Functions and Solution Mappings*. Springer, 2009.