A Predictive Motion Planner for Guidance of Autonomous UAV Systems

Peter T. Jardine and Sidney Givigi
Department of Electrical and Computer Engineering
Royal Military College of Canada
Kingston, ON., Canada
Email: peter.jardine@rmc.ca, sidney.givigi@rmc.ca

Abstract— **This paper investigates Unmanned Aerial Vehicle (UAV) systems motion planning for ground attack missions involving enemy defenses. The UAV dynamics are modeled as a unicycle, linearized using dynamic extension and expanded over a finite prediction horizon as a piece-wise affine function. The motion planning problem is then formulated as a constrained, convex minimization in the form of Linear Quadratic Model Predictive Control (LQMPC). Avoidance of enemy defenses is achieved using linear inequality constraints. The design is tested in a simulated ground attack mission involving a layered enemy defense system using MATLAB. Preliminary results demonstrate the feasibility of using LQMPC to guide a UAV in ground attack missions involving complex enemy defenses.**

I. Introduction

This paper deals with UAV systems motion planning for ground attack missions involving enemy defenses. Currently, these types of UAV operations typically involve a significant amount of human control, particularly for high-level decision making [1],[2]. This presents a number of limitations, including the high bandwidth used for command and control and the additional time required to include humans in the decision loop [3]. Overcoming these limitations motivates research into autonomous UAVs capable of making on-board decisions in real-time. An essential function of autonomous vehicles is the ability to make informed decisions about motion planning [4]. Optimal planning techniques should anticipate the future consequences of control actions while making the most efficient use of limited resources [5], particularly for UAV attack scenarios, where prolonged exposure to enemy defenses could result in mission failure [3].

Recently, Model Predictive Control (MPC) has emerged as a popular motion planning strategy because of its ability to provide optimized control policies for multi-variable linear or non-linear systems while adhering to state and input constraints [6]. The main limitation of MPC is the computation time required during optimization, which can stymie attempts at real-time implementation [7]. However, given certain assumptions, efficient optimization techniques can be applied to MPC problems and possibly implemented in real-time [8].

One method for achieving tractable solution was pioneered in [8] and [9]. Their approach presents the set of future states as a piece-wise affine function and formulates the optimization as a multi-parametric, convex quadratic program in the form of a Linear Quadratic MPC (LQMPC) [10]. Given linear system dynamics, linear inequality constraints and a quadratic objective function, there exist computationally efficient algorithms for solving these types of problems [11].

Various approaches have been taken to incorporate obstacle avoidance in the MPC framework. These can be considered in two broad categories: those which discourage collisions as part the objective function [12]; or those which represent obstacles as a constraint on the system states [13]. Using linear, time-varying system dynamics, [14] created linear inequality constraints from points tangent to the Gaussian distributions of uncertainty around obstacles. These constraints are updated as the system evolves. As a result, the set of feasible solutions takes the form of a convex polytope and can be efficiently solved as a convex optimization problem.

Motion planning research for UAV ground attack scenarios has focused mainly on the problem of coordinating multiple UAVs [15]. Few of the proposed solutions apply the concepts of MPC to provide an optimized control policy for ground attack scenarios while incorporating enemy defenses.

In [3], the authors present a generic battlefield scenario based on layered defenses protecting a stationary ground target. The Enemy Defense System Model consists of three concentric circles centered on a high-value target. Each layer is characterized by the presence of different weapons platforms, such as static anti-aircraft missile batteries or mobile anti-aircraft guns. The authors use Dubins paths to determine optimal grouping patterns and routes for UAV strike teams.

This research formulates the ground attack motion planning problem described in [3] as a convex, quadratic programming optimization in the form of LQMPC. Avoidance of enemy defenses is achieved by imposing linear inequality constraints on the system states, thereby creating a region of feasible solutions that is convex. This allows for the efficient computation of optimized control policies. The performance of the planning algorithm is tested in realistic MATLAB simulations. It is envisioned that such a solution could be ported to a UAV as part of its guidance system.

Section II of this paper describes the components of the UAV system and the relationships among them. Section III states the problem in terms of vehicle dynamics, obstacle dynamics, and mission objectives. Section IV formulates target tracking and enemy avoidance as the constrained, convex optimization of a piece-wise affine function. Simulation results are presented in Section V. Conclusions and recom-

978-1-4673-9519-9/16/$31.00 ©2016 Crown

mendations for future work are presented in Section VI.

II. UAV System

Fig. 1 provides an illustration of the UAV system components and the relationships among them. The focus of this paper is on the guidance system, which, as shown in Fig. 1, can be placed on board the UAV platform itself. The guidance system uses information collected from sensors to develop a motion plan (trajectory) to the target while adhering to constraints. This motion plan is fed to a low-level controller to execute movement of the actual UAV control surfaces. Commands can be sent to the UAV from a ground station via the on-board communications system.

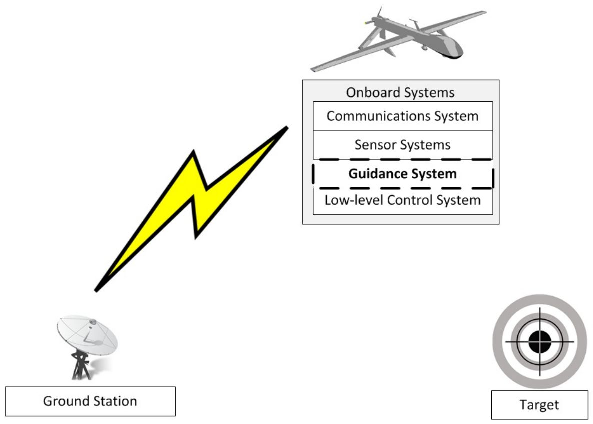

Fig. 1. UAV system components

A human operator normally makes decisions about target selection, route planning and maneuvering. Notice this would require direct communication with the UAV platform through the ground station. In a rapidly changing environment, these decisions would suffer from a latency that could jeopardize the mission. Therefore, it is desirable that guidance be performed on-board of the UAV.

III. Problem Statement

This section develops a motion planning problem statement which encompasses vehicle dynamics, obstacle dynamics and mission objectives. Section III-A describes the method of dynamic extension used to develop a linear approximation for UAV dynamics. Section III-B describes the dynamics for mobile enemy defenses. Section III-C provides a mathematical representation for the ground attack mission considered in this research in the form of an objective function.

A. Vehicle Dynamics

Consider a UAV at constant altitude with position coordinates x, y, and θ, forward velocity, v, angular velocity, ω, and dynamics governed by (1):

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}v\cos\theta\\ v\sin\theta\\ \omega\end{bmatrix} \tag{1}$$

where $\dot{x}, \dot{y}$ and $\dot{\theta}$ are the time rate of change in position. Defining a vector, $z = \begin{bmatrix}x & y\end{bmatrix}^T$, we want the error between z and a desired vector, z_d to approach 0 as $t \to \infty$. The second derivative, $\ddot{z}$ is then computed as:

$$\ddot{z} = \begin{bmatrix}\cos\theta & -v\sin\theta\\ \sin\theta & v\cos\theta\end{bmatrix}\begin{bmatrix}\dot{v}\\ \omega\end{bmatrix} \tag{2}$$

Assuming $v \neq 0$ and letting the input vector, $u = \begin{bmatrix}u_1 & u_2\end{bmatrix}^T = \ddot{z}$, (2) is rearranged and combined with (1) to obtain a new model for the vehicle as:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \dot{v}\end{bmatrix} = \begin{bmatrix}v\cos\theta\\ v\sin\theta\\ -u_1\frac{\sin\theta}{v} + u_2\frac{\cos\theta}{v}\\ u_1\cos\theta + u_2\sin\theta\end{bmatrix} \tag{3}$$

Given the change of variables, $\gamma_1 = x$, $\gamma_2 = y$, $\gamma_3 = \dot{x}$, and $\gamma_4 = \dot{y}$, (3) is expressed as the following linear state space model:

$$\begin{bmatrix}\dot{\gamma}_1\\ \dot{\gamma}_2\\ \dot{\gamma}_3\\ \dot{\gamma}_4\end{bmatrix} = \begin{bmatrix}0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\gamma_1\\ \gamma_2\\ \gamma_3\\ \gamma_4\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\\ 1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix} \tag{4}$$

The discrete form of (4) with sample time, dt is as follows:

$$\underbrace{\begin{bmatrix}\gamma_{1,k+1}\\ \gamma_{2,k+1}\\ \gamma_{3,k+1}\\ \gamma_{4,k+1}\end{bmatrix}}_{\gamma_{k+1}} = \underbrace{\begin{bmatrix}1 & 0 & dt & 0\\ 0 & 1 & 0 & dt\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}}_{A}\underbrace{\begin{bmatrix}\gamma_{1,k}\\ \gamma_{2,k}\\ \gamma_{3,k}\\ \gamma_{4,k}\end{bmatrix}}_{\gamma_k} + \underbrace{\begin{bmatrix}dt^2 & 0\\ 0 & dt^2\\ dt & 0\\ 0 & dt\end{bmatrix}}_{B}\underbrace{\begin{bmatrix}u_{1,k}\\ u_{2,k}\end{bmatrix}}_{u_k} \tag{5}$$

where and A and B are the system and input matrices and γ_k and u_k are the state and input vectors at timestep, k. The vehicle position is measured directly according to the measurement model:

$$\psi_k = \underbrace{\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}}_{C}\gamma_k \tag{6}$$

where ψ_k is the measured output vector and C is the measurement matrix.

In order to obtain the remaining states, a linear quadratic estimator (Kalman Filter) is used. For a detailed description of Kalman filter techniques and their application to navigation, refer to [16] and [17].

B. Enemy Dynamics

Mobile enemy defense platforms are assumed to have dynamics as described in (5). When the UAV enters the appropriate defensive layer, mobile platforms are able to begin tracking the UAV using a simple proportional feedback controller of the form:

$$u_e = K_e(\gamma_v - \gamma_e) \tag{7}$$

where γ_v and γ_e are the states of the UAV and mobile platform, u_e is the input vector and K_e is the feedback gain.

The dynamics of the enemy platforms are constrained by a maximum velocity, $v_{e,max}$.

C. Mission Objectives

Using the battlefield scenarios described in [3] as a benchmark, Fig. 2 illustrates the ground attack scenario considered in this research. The UAV must pass through all layers of the enemy defenses to fly over and destroy the target. Each weapon system has a set of parameters that describe its range and mobility. When the UAV comes within radius *r₁* of the target, the first layer of defense is activated. At this point the mobile gun platforms can start moving towards the UAV. Once the UAV is within range, they can begin firing at the UAV.

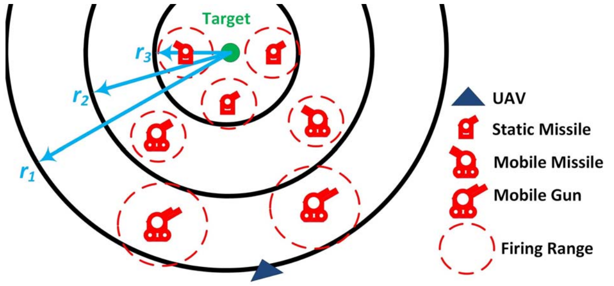

Fig. 2. Battlefield scenario with enemy anti-aircraft weapons defending a high-value target (not to scale)

Similarly, when the UAV comes within radius *r₂* of the target, the second layer of defenses is activated. At this point the mobile missile platforms can start moving towards the UAV and fire when in range. The final layer of defenses is activated when the UAV comes within radius *r₃* of the target. At this point any static missile platforms can fire on the UAV when it flies within range. In all cases, the more time the UAV spends within range of enemy defenses, the greater the likelihood it has of being destroyed.

The objective function, J which drives the measured output, ψ_k towards the target at position $r(t)$ is then expressed as:

$$J(\gamma_{1:N}, u_{1:N}) = \sum_{k=0}^{N-1} [(r(t) - \psi_{k+1})^T Q (r(t) - \psi_{k+1}) + \Delta u_k^T R \Delta u_k + \gamma_N^T P \gamma_N] \tag{8}$$

where $\gamma_{1:N}$ and $u_{1:N}$ are the states and inputs for each time step, k over finite horizon, N and Δu_k is the vector representing the change in input between successive time steps. Minimizing (8) means minimizing the distance between the UAV and the target. Avoidance of enemy defenses is achieved by imposing state constraints on the optimization as described in the following section. Q and R are diagonal weighting matrices on the stage cost and P is the diagonal weighting matrix on the terminal cost.

IV. Solution

This section formulates the problem in Section III as a convex, quadratic programming optimization in the form of LQMPC. Section IV-A presents the evolution of system dynamics as a piece-wise affine function and incorporates this into the quadratic objective function to achieve target tracking. Section IV-B shows how the range of enemy defenses can be avoided using linear inequality constraints. Section IV-C briefly discusses stability. With linear system dynamics and a quadratic objective function constrained by linear inequalities, the MPC optimization is convex and hence can be solved efficiently.

A. Target Tracking

In [10], the author provides a framework for implementing a quadratic program to track a reference signal. Assuming linear system dynamics, the control inputs can be redefined in terms of input increments as:

$$\Delta u_k = u_k - u_{k-1} \tag{9}$$

and the standard linear state space model extended to include the new variable $\gamma_{u,k} = u_{k-1}$:

$$\underbrace{\begin{bmatrix} \gamma_{k+1} \\ \gamma_{u,k+1} \end{bmatrix}}_{\bar{\gamma}_{k+1}} = \underbrace{\begin{bmatrix} A & B \\ 0 & I \end{bmatrix}}_{\bar{A}} \underbrace{\begin{bmatrix} \gamma_k \\ \gamma_{u,k} \end{bmatrix}}_{\bar{\gamma}_k} + \underbrace{\begin{bmatrix} B \\ I \end{bmatrix}}_{\bar{B}} \Delta u_k$$

where $\bar{\gamma}_k$ is the new state vector including γ_k and $\gamma_{u,k}$. The measurement model is also extended to include this additional state:

$$\bar{y}_k = \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{\bar{C}} \bar{\gamma}_k$$

This relationship is simplified using a new notation:

$$\bar{\gamma}_{k+1} = \bar{A}\bar{\gamma}_k + \bar{B}\Delta u_k \tag{10}$$

and

$$\bar{y}_k = \bar{C}\bar{\gamma}_k \tag{11}$$

where $\bar{y}_k$ indicates a measurement derived from the extended state vector, $\bar{\gamma}_k$. By expanding the system dynamics and measurement model as a piece-wise affine function N time steps into the future, we obtain the set of future measurements in terms of the initial states and the control policy as follows:

$$\underbrace{\begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \\ \bar{y}_3 \\ \vdots \\ \bar{y}_N \end{bmatrix}}_{\bar{Y}} = \underbrace{\begin{bmatrix} \bar{C}\bar{B} & 0 & 0 & \dots & 0 \\ \bar{C}\bar{A}\bar{B} & \bar{C}\bar{B} & 0 & \dots & 0 \\ \bar{C}\bar{A}^2\bar{B} & \bar{C}\bar{A}\bar{B} & \bar{C}\bar{B} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{C}\bar{A}^{N-1}\bar{B} & \bar{C}\bar{A}^{N-2}\bar{B} & \bar{C}\bar{A}^{N-3}\bar{B} & \dots & \bar{C}\bar{B} \end{bmatrix}}_{\bar{S}} \underbrace{\begin{bmatrix} \Delta u_0 \\ \Delta u_1 \\ \Delta u_2 \\ \vdots \\ \Delta u_{N-1} \end{bmatrix}}_{\Delta U} + \underbrace{\begin{bmatrix} \bar{C}\bar{A} \\ \bar{C}\bar{A}^2 \\ \bar{C}\bar{A}^3 \\ \vdots \\ \bar{C}\bar{A}^N \end{bmatrix}}_{\bar{T}} \bar{\gamma}_0$$

or, simply:

$$\bar{Y} = \bar{S}\Delta U + \bar{T}\bar{\gamma}_0 \tag{12}$$

where $\bar{Y}$ and $\overline{\Delta U}$ are the set of output and input vectors and $\bar{S}$ and $\bar{T}$ are derived from the expansion of $\bar{A}$ and $\bar{B}$ over N. By substituting (12) into (8) and rearranging we find the tracking problem can be solved as the quadratic programming optimization of:

$$\begin{aligned} J(\bar{\gamma}_{1:N}, u_{1:N}) =& \Delta U^T \underbrace{[\bar{S}^T \bar{Q} \bar{S} + \bar{R}]}_{H} \Delta U \\ &+ 2 \underbrace{[\bar{\gamma}_0^T \bar{T}^T \bar{Q} \bar{S} - r(t)^T \bar{Q} \bar{S}]}_{f} \Delta U \\ &+ \gamma_N^T P \gamma_N \end{aligned} \tag{13}$$

subject to constraints

$$\begin{aligned} M_\gamma \Gamma \leq f_\gamma \\ M_u U \leq f_u \end{aligned} \tag{14}$$

where Γ and U are the stacked state and input vectors over the prediction horizon, M_γ and M_u are the constraint matrices, f_γ and f_u are the constraint vectors and $\bar{Q}$ and $\bar{R}$ are the block diagonal matrices constructed as follows:

$$\bar{Q} = \begin{bmatrix} Q_1 & 0 & 0 & 0 & 0 \\ 0 & Q_2 & 0 & 0 & 0 \\ 0 & 0 & Q_3 & 0 & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & & 0 \\ 0 & 0 & 0 & 0 & Q_N \end{bmatrix}, \bar{R} = \begin{bmatrix} R_1 & 0 & 0 & 0 & 0 \\ 0 & R_2 & 0 & 0 & 0 \\ 0 & 0 & R_3 & 0 & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & & 0 \\ 0 & 0 & 0 & 0 & R_N \end{bmatrix}$$

For this application, only the first optimized input is selected to drive the system. At the next time step, a whole new plan is developed over horizon, N. This approach is sometimes referred to as Receding Horizon Control [18].

B. Enemy Avoidance

Similar to the approach used in [14], the range of enemy defenses are considered as obstacles using linear inequality constraints. Normally, imposing a constraint on the system states based on the location of obstacles creates a space of feasible solutions that is non-convex [14]. Therefore, the optimization in Section IV-A would become non-convex and difficult to solve. To conserve a convex region of feasible solutions, enemy avoidance is formulated as linear inequality constraints placed along special points on the outer range of the enemy defenses.

Fig. 3 illustrates how linear inequality constraints can be used to avoid enemy defenses. Given the position of the UAV (x_v, y_v) and position of an enemy platform (x_e, y_e) with an effective firing range of radius r_e, we define the straight line with length l_e and slope m_e between the center of mass of the two objects. The point (x_p, y_p) lies on this line at a distance r_e from the enemy defense. The linear inequality constraint is defined by the line with slope perpendicular to m_e, tangent to the circle drawn around the enemy platform of radius r_e and intersecting (x_p, y_p) (see Fig. 3).

Given m_e computed as follows:

$$m_e = \frac{y_e - y_v}{x_e - x_v} \tag{15}$$

then $m_e' = -1/m_e$ is the slope perpendicular to m_e. The angle θ_e between the vehicle and the enemy platform with respect to the x-axis is computed as:

$$\theta_e = \arctan(m_e) \tag{16}$$

Assuming the distance between the two objects is l_e, we use θ_e to find (x_p, y_p) as follows:

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} (l_e - r_e) \cos\theta_e + x_v \\ (l_e - r_e) \sin\theta_e + y_v \end{bmatrix} \tag{17}$$

Therefore, for each obstacle at a given timestep, a linear matrix inequality can be used to constrain the UAV states to a region above or below the line with slope m' intersecting (x_p, y_p) as follows:

$$\begin{cases} -m_e'\bar{\gamma} + \bar{\gamma} \leq -m_e' x_p + y_p, & \text{if } \pi \geq \theta_e > 0 \\ \\ m_e'\bar{\gamma} - \bar{\gamma} \leq m_e' x_p - y_p, & \text{if } \pi < \theta_e \leq 2\pi \end{cases} \tag{18}$$

which is a relationship conditional on the value of θ_e. This linear constraint is assumed constant for the entire prediction horizon. Once the first optimized control input is executed, the UAV (and possibly the enemy platform) will have moved. An updated set of linear constraints is then developed and used for the next optimization. This process continues for each timestep until the target is reached.

As shown by constraint 2 in Fig. 3, certain situations may find the enemy platform placed directly between the UAV and target. In cases like this, the inequality constraint would be perpendicular to the optimum flight path and could drive the UAV directly towards the enemy platform. To reduce the likelihood of this occurring, a special condition is used which rotates the constraint by θ_d around the obstacle, encouraging the UAV to favor one side. This is accomplished by computing the inequality constraint using a new point (x_p', y_p') determined by the following rotational transformation:

$$\begin{bmatrix} x_p' \\ y_p' \end{bmatrix} = \begin{bmatrix} \cos\theta_d & -\sin\theta_d \\ \sin\theta_d & \cos\theta_d \end{bmatrix} \begin{bmatrix} x_p - x_e \\ y_p - y_e \end{bmatrix} + \begin{bmatrix} x_e \\ y_e \end{bmatrix} \tag{19}$$

For the simulations described in this paper, this transformation was only applied when the angle between the UAV heading and θ_e were within $\frac{\pi}{8}$ rads of each other.

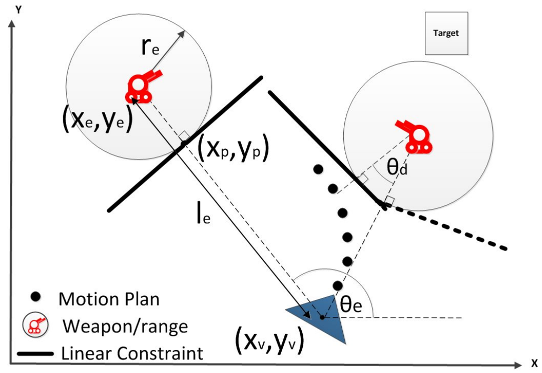

Fig. 3. Illustration of linear inequality constraints for enemy avoidance (not to scale)

C. Stability

Typically, MPC stability is established by expressing the optimal cost $J^*(\gamma_{1:N}, u_{1:N})$ as a Lyapunov function [19] and placing the terminal state, γ_N inside a terminal set, $\mathbb{X}_T$.

Theorem 1: Considering quadratic objective function (8), linear model (10) and linear, matrix inequality constraints (14) and (18), when formulated as an MPC, the system will be exponentially stable.

Proof: If P is selected as the positive-definite solution of the Discrete Algebraic Riccati Equation:

$$P = Q + \bar{A}^T(P - P\bar{B}(R + \bar{B}^T P\bar{B})^{-1}\bar{B}^T P)\bar{B} \qquad (20)$$

and the following assumptions are true [20]:

- $\mathbb{X}_T$ is a closed set containing the origin and all states inside $\mathbb{X}_T$ satisfy the state constraints.
- The control input constraints are satisfied inside $\mathbb{X}_T$.
- $\mathbb{X}_T$ is positively invariant.

then it follows according to Theorem 2.1 in [21] that the terminal objective function is a Control-Lyapunov function and the system is exponentially stable. ■

V. Simulation and Results

The motion planning algorithm described in Section IV was implemented in MATLAB. The CVX modeling system (version 2.1) was used to provide convex optimization solutions [22]. Fig. 4 shows seven enemy defenses placed uniformly between the initial UAV position at $(1000, 0)$ and the target at $(3000, 3000)$. The radius of each defense layer was 3000 m, 2000 m, and 700 m centered on the target. The outer defense layer consisted of two enemy platforms, each with an effective range of 400 m and maximum velocity of 20 m/s. The middle defense layer consisted of two mobile enemy platforms, each with an effective range of 250 m and maximum velocity of 20 m/s. All mobile platforms were assumed to be capable of firing while in motion and pursue the UAV according to the dynamics described in Section III-B. The inner defense layer consisted of three static enemy platforms, each with a range of 250 m. These parameters were based on the approximate capabilities of shoulder-mounted rocket propelled grenade launchers and small arms described in [23].

In order to partially account for the dynamics of the mobile platforms, small buffer zones were created around each obstacle. The values of the buffer zones were based on the distance an enemy platform could travel in the given sample time. For example, given a maximum velocity of 20 m/s and sample time 0.5s, it is feasible that the enemy platform could be 10 m closer than expected after one timestep. Given a buffer zone of 10 m, the new radius would be computed as $(r'_e = r_e + 10\ m)$.

The UAV was given a maximum velocity of 60 m/s. A sampling time of 0.5 s and prediction horizon of 10 were used. The weighting matrices, Q and R were given equal values of the identity matrix. In Fig. 5 we see the simulated UAV successfully navigate to the target.

Fig. 6 illustrates the observed distance from enemy defenses throughout the simulation. Here we see the UAV stay outside the range of enemy defenses at all times. The closest pass was within 5 m of a static defense platform in a congested area near the target.

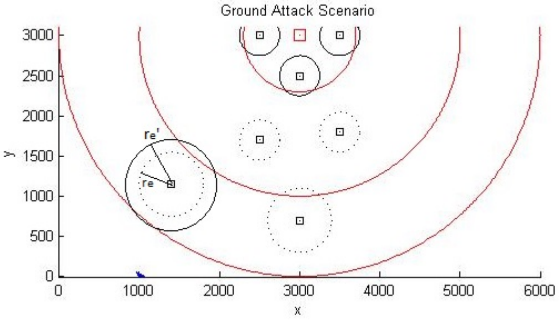

Fig. 4. Placement of target and enemy defenses for simulation

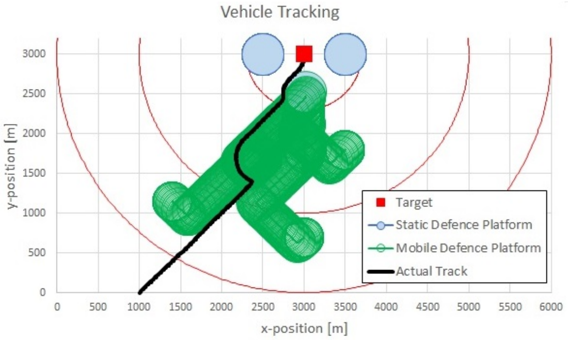

Fig. 5. Simulation results show UAV successfully navigate around enemy defenses to target

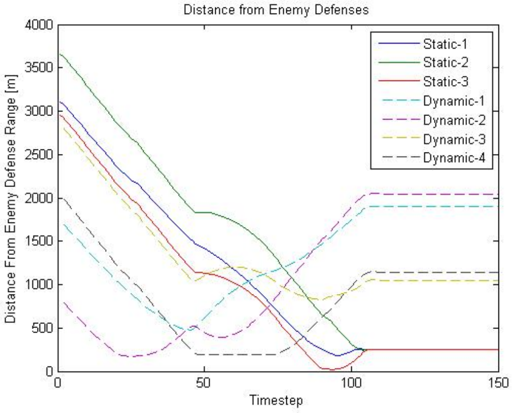

Fig. 6. Simulation results show UAV stay outside the range of enemy defenses

Fig. 7 shows the overall cost decreasing throughout the simulation. Minor increases in cost were observed when maneuvering around enemy defenses, particularly at around timestep 25.

Fig. 8 shows the acceleration inputs at each timestep. Here we see peaks and valleys where aggressive maneuvers were required to navigate around enemy defenses. Some oscillation in inputs were observed between timesteps 50 and 75. These oscillations were caused by the dynamics of mobile defenses. Since the defense platforms were assumed static during each prediction, the UAV could not anticipate and plan around the path of the mobile defenses. Therefore,

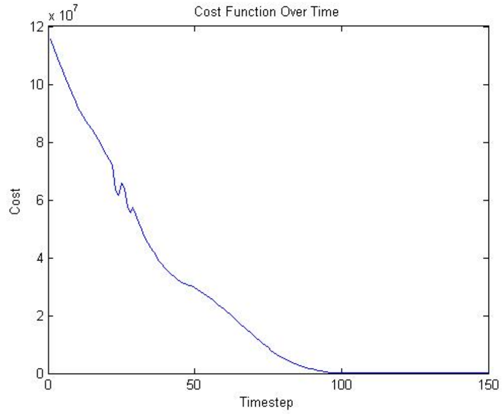

Fig. 7. Simulation results show overall cost decreases with time

constant corrections were required when passing near them. Addressing this issue will be a focus of future research.

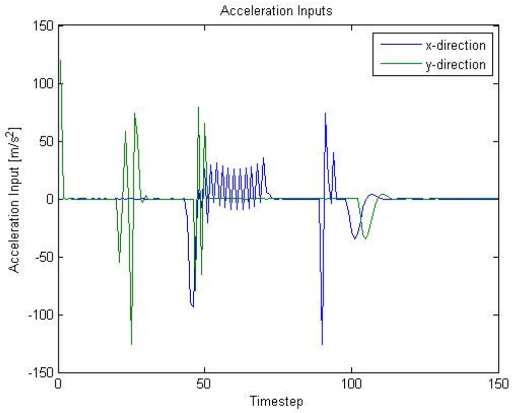

Fig. 8. Simulation results show control inputs while avoiding enemy defenses

VI. Conclusion

Motivated by the desire to increase autonomy on the battlefield, this paper showed that UAV ground attack missions can be formulated as a convex, quadratic programming optimization in the form of LQMPC. Avoidance of mobile enemy defenses can be treated as linear inequality constraints, which make the set of feasible solutions convex and hence capable of being solved efficiently.

One limitation on this formulation is the assumption that enemy platforms are stationary during the prediction step. Future research will focus on incorporating anticipated enemy dynamics while maintaining a region of convex solutions. An important consideration for this approach will be the impact this has on computation time. Stability in the presence of process noise and measurement uncertainty should also be investigated. The effects of real-time constraints should be considered prior to implementation on an actual system.

References

[1] R. Sparrow, "Predators or plowshares? Arms control of robotic weapons," *IEEE Technology and Society Magazine*, vol. 28, no. 1, pp. 25–29, Mar 2009.

[2] S. Giese, D. Carr, and J. Chahl, "Implications for unmanned systems research of military UAV mishap statistics," in *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*, Jun 2013, pp. 1191–1196.

[3] M. Suresh and D. Ghose, "UAV grouping and coordination tactics for ground attack missions," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 48, no. 1, pp. 673–692, Jan 2012.

[4] T. Howard, M. Pivtoraiko, R. Knepper, and A. Kelly, "Model-predictive motion planning: Several key developments for autonomous mobile robots," *IEEE Robotics Automation Magazine*, vol. 21, no. 1, pp. 64–73, Mar 2014.

[5] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, Jun 2005.

[6] J. Yan and R. R. Bitmead, "Incorporating state estimation into model predictive control and its application to network traffic control," *Automatica*, vol. 41, no. 4, pp. 595 – 604, Apr 2005.

[7] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, Mar 2010.

[8] A. Bemporad and C. Filippi, "Suboptimal explicit MPC via approximate multiparametric quadratic programming," in *Proc. of the 40th IEEE Conf. on Decision and Control*, vol. 5, Dec 2001, pp. 4851–4856 vol.5.

[9] P. Tndel, T. A. Johansen, and A. Bemporad, "An algorithm for multi-parametric quadratic programming and explicit MPC solutions," *Automatica*, vol. 39, no. 3, pp. 489 – 497, Mar 2003.

[10] A. Bemporad, "Model predictive control: Basic concepts," 2009. [Online]. Available: www.seas.upenn.edu/∼ese680/papers/IntroductionMPC.pdf

[11] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[12] M. Boccardo, M. Egerstedt, and Y. Wardi, "Obstacle avoidance for mobile robots using switching surface optimization," in *Proc. of Int. Fed. of Automatic ControlWorld Congress*, Jul 2005.

[13] J. Frasch, A. Gray, M. Zanon, H. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," in *Proc. of the European Control Conf. (ECC)*, Jul 2013, pp. 4136–4141.

[14] M. Mousavi, Z. Heshmati, and B. Moshiri, "LTV-MPC based path planning of an autonomous vehicle via convex optimization," in *Proc. of the 21st Iranian Conf. on Electrical Engineering (ICEE)*, May 2013, pp. 1–7.

[15] A. Hafez, M. Iskandarani, S. Givigi, S. Yousefi, C. A. Rabbath, and A. Beaulieu, "Using linear model predictive control via feedback linearization for dynamic encirclement," in *Proc. of the American Control Conf.*, Jun 2014, pp. 3868–3873.

[16] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[17] A. Noureldin, T. Karamat, and J. Georgy. Springer London, 2012 booktitle=Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration.

[18] N. Du Toit and J. Burdick, "Robot motion planning in dynamic, uncertain environments," *IEEE Trans. on Robotics*, vol. 28, no. 1, pp. 101–115, Feb 2012.

[19] J. A. Rossiter, *Model-Based Predictive Control: A Practical Approach*, 1st ed., R. H. Bishop, Ed. Boca Raton, FL: CRC Press LLC, 2004.

[20] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789 – 814, Jun 2000.

[21] M. Balandat, "Constrained robust optimal trajectory tracking: Model predictive control approaches," Diploma Thesis, Technische Universitt Darmstadt, 2010.

[22] M. Grant and S. Boyd. (2014, Jun) Cvx: Matlab software for disciplined convex programming. [Online]. Available: http://cvxr.com/cvx/

[23] J. Pike. (2014, Aug) Global security database. [Online]. Available: http://www.globalsecurity.org/org/index.html