Model Predictive Control for Target Tracking in 3D with a DownwardFacing Camera Equipped Fixed Wing Aerial Vehicle

by

Pravin Mali, Arun Kumar Singh, Madhava Krishna, Sujit P.B.

Report No: IIIT/TR/2020/-1

Centre for Robotics
International Institute of Information Technology
Hyderabad - 500 032, INDIA
August 2020

Model Predictive Control for Target Tracking in 3D with a Downward Facing Camera Equipped Fixed Wing Aerial Vehicle

Pravin Mali[1], Arun Kumar Singh[2], Madhav Krishna[1] and P.B. Sujit[3]

***Abstract*— In this paper, we consider the problem of tracking a ground vehicle with a fixed-wing aerial vehicle (FWV) equipped with a downward-facing camera. The complexity of the problem stems from the highly nonlinear kinematics of the FWV and the stall speed constraint. We propose a Model Predictive Control (MPC) approach for this problem that has two main contributions. Firstly, we model the tracking requirement through a novel constraint function that relates FWV's position and orientation to the field of view of the camera. Secondly, we make a case for reformulating the underlying optimization of the MPC as an unconstrained problem and solving it through the state of the art gradient descent variants like ADAM and RMSProp. Specifically, we show the real-time performance of this optimizer while achieving good tracking performance under various kinematic constraints. We validate our MPC through extensive simulations, specifically highlighting the 3D spiral-like trajectories obtained for the FWV when tracking a slow-moving ground vehicle. We also present a quantitative analysis of the efficacy of the different gradient descent variants.**

I. Introduction

A. Problem Description

This paper deals with the problem of persistent tracking, wherein a fixed-wing unmanned aerial vehicle (FWV) needs to always maintain a moving target vehicle within its field of view (FOV) as shown in Fig. 1. Persistent tracking is often accomplished with a fixed-wing aerial vehicle because of its ability to leverage the wind profile to perform long-range flights with limited on-board power. However, the benefits of a FWV in this application comes at the expense of increased complexity in motion planning and control. For example, due to stall speed constraint a FWV cannot reduce its speed beyond a certain limit in order to sustain flight. This, in turn, means that when tracking a slow-moving target vehicle, it has to perform complex maneuvers (spiral-like) to maintain tracking.

A large volume of existing works [1], [2], [3], [4], [5], [6], [7] formulate persistent target tracking as a problem of ensuring some relative distance and line-of-sight between the FWV and the target vehicle. The underlying implicit assumption here is that for the achieved relative position and orientation, the FOV is either large enough to track the vehicle or can be adapted appropriately by changing the pan-tilt of the camera. This assumption simplifies the problem by making the FOV at any given instant independent of the orientation of the FWV. In sharp contrast, the problem that we consider is more constrained and challenging. We consider a downward-facing camera with limited FOV and which is connected rigidly to the FWV. As a result, for the given relative distance between the FWV and the target vehicle, the FOV can be severely affected by the orientation of the former. Furthermore, the evolution of relative position and FWV orientation is itself coupled in a highly non-linear manner through the so-called non-holonomic kinematics. Our problem formulation is similar in spirit to [8], designed for keeping static objects within the FOV of a multi-rotor drone. This class of vehicles has simpler kinematics in addition to the ability to hover in-place, leading to a substantially simpler problem than that considered in this paper.

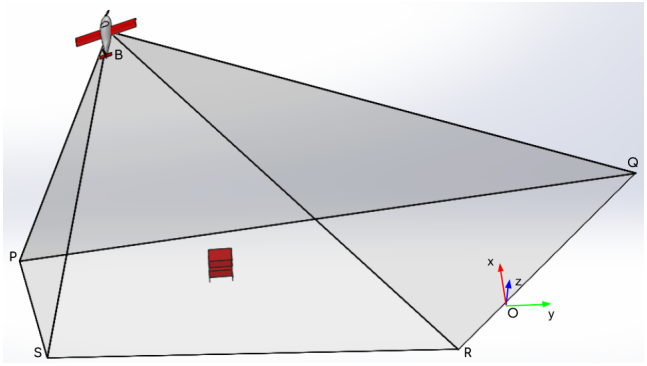

Fig. 1. A FWV tracking a ground vehicle. The FOV of the FWV camera is modeled as a prism and thus its projection on the ground is a quadrilateral.

Contribution: Our solution approach is based on Model Predictive Control (MPC) which entails solving an optimization problem in a receding horizon manner. In this context, we have two main contributions. Firstly, we formulate the persistent tracking as a simple constraint that relates FWV's orientation, its relative distance with the target vehicle with the camera FOV. Secondly, we reformulate our constrained optimization as a non-smooth unconstrained problem and solve it through state-of-the-art gradient descent variants like ADAM [9] and RMSProp [10]. Although these optimizers are quite popular in deep learning, their use in MPC has been very limited. We are only aware of [11] that apply the optimizer to off-line motion planning. In contrast, we believe the low per-iteration complexity of these optimizers makes them well suited to an MPC setup, wherein due to real-time constraints, one can only afford to perform few iterations of the underlying optimization. These optimizers also do not require any computation of matrix factorization or inverse making them suitable for the implementation of embedded hardware [12]. Furthermore, matrix multiplication can be

1. Robotics Research Center, IIIT-Hyderabad, India. 2. Institute of Technology, University of Tartu. 3. IISER Bhopal, Bhopal, India. Emails: pravin.mali@research.iiit.ac.in, arun.singh@ut.ee, mkrishn@iiit.ac.in, sujit@iiserb.ac.in

easily accelerated on GPUs making our MPC approach well suited for hardware like Jetson Nano[1] that have a low-end CPU but reasonably good GPUs. We validate our formulation on several simulation experiments, specifically highlighting the 3D spiral-like trajectories of FWV. We also present a quantitative comparison of the performance of ADAM and RMSProp in terms of constraint satisfaction and tracking efficiency.

B. Related Work

The target tracking problem has been of interest for over two decades in the robotics literature. One technique to track targets is to develop guidance laws [1], [2], [3], [4], [5], [6], [7]. The guidance laws take the target model and track it while ensuring the target is within a radius of r units which mimics a FOV constraint. however, actual FOV constraints due to the FWV motion is not considered.

There have been several efforts in developing vehicle controllers for a single UAV tracking targets in urban environments [13], [14], [15], [16], [17], [18], [19]. Zhao *et al.* [18] developed a vision algorithm based on YOLO to detect a target in an urban environment. A simple proportional controller is used to track the target. Wantabi and Fabiani [15] developed an optimal guidance framework for tracking a target in an urban environment. However, optimal guidance requires a target model and is also computationally intensive. Semsch *et al.* [16] converted the surveillance problem to obtain information at certain locations in an urban environment into an art gallery problem taking visibility constraints into account and then use a TSP-based approach to find the path for the UAV. However, in this case, the target tracking aspects are not considered. Ramirez *et al.* [13] developed an information theoretic planner that has an estimate of the target and this estimate is updated based on ground sensors and UAV camera sensor. Kim and Crassidis [14] assign circular paths to maximize the visibility of the targets and the decision to changes these circular paths is carried out online. Wu *et al.* [17] developed an improved whale optimization framework to determine paths for the UAV to maximize the energy obtained by the solar panels. However, they do not consider the target tracking aspect. Theodorakopoulos and Lacroix [19] developed an iterative optimizing method to track the target. A set of trajectories are predicted and evaluated. Based on the cost of these trajectories taking the visibility constraint and the obstacle avoidance into account, a path is determined and given to the UAV for tracking. However, the paths obtained are generated through a heuristic.

[1] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

II. Formulating Target Tracking

A. FWV Kinematics

The discrete-time 3D kinematics of a FWV is given by the following equations.

$$\mathbf{x}_{t+1} = \mathbf{x}_0 + \sum_{t=0}^{t=n-1} \mathbf{f}(\mathbf{u}_t), \tag{1}$$

where the states $\mathbf{x}_t = (x_t, y_t, z_t, \psi_t)$ consists of the 3D position and the heading angle of FWV. The control inputs $\mathbf{u}_t = (v_t, \gamma_t, \psi_t)$ are taken as the airspeed, the flight path angle and the bank-angle. Due to the specific nature of the kinematics, the airspeed $v_t \geq v_{min}$ should always be ensured for the FWV to sustain flight. The motion model $\mathbf{f}$ is defined as the following, where g refers to the acceleration due to gravity.

$$\mathbf{f}(\mathbf{u}_t) = \begin{bmatrix} v_t \cos(\psi_t) \cos\gamma_t \Delta t \\ v_t \sin(\psi_t) \cos\gamma_t \Delta t \\ -v_t \sin\gamma_t \Delta t \\ \frac{g}{v_t} \tan\phi_t \Delta t \end{bmatrix} \tag{2}$$

B. MPC

We assume that the target vehicle is moving in the $x-y$ plane and we have an algorithm in place for predicting its trajectory over a time horizon. With this context and considering a planning horizon of $t \in [0\ n]$, the optimization underlying our MPC can be represented through the following equations.

$$\min_{\mathbf{u}_t} c(x_n, y_n) + \sum_t \|\mathbf{u}_t\|_2^2 \tag{3}$$

$$\mathbf{g}_{bounds}(\mathbf{u}_t) \leq 0, \forall t \tag{4}$$

$$\mathbf{g}_{fov}(\mathbf{u}_t) \leq 0, \forall t \tag{5}$$

$$\tag{6}$$

$$c(x_n, y_n) = (x_n - x_n^{target})^2 + (y_n - y_n^{target})^2 \tag{7}$$

Where, $(x_n^{target}, y_n^{target})$ is the position of the target vehicle in the $x-y$ plane. The cost function (3) has two parts. The first term aims to minimize the distance between the FWV and the tracked vehicle at the end of the planning horizon. This, in turn, encourages the FWV to bring the tracked vehicle to the center of its FOV. The second term in the cost function ensures the regularization of the control input. The constraints (4) represent the kinematic bounds on the velocity, flight path angle, the bank angle, and their derivatives. The inequalities (5) represent the FOV constraints to ensure that the tracked vehicle can be seen by the downward-facing camera of the FWV at all time instants.

Constrained Optimization (3)-(5) is highly non-linear and non-convex. Nevertheless, techniques like sequential quadratic programming implemented in software like Scipy-SLSQP [20] can be used to obtain a locally optimal solution. But as mentioned earlier, we want to adopt a technique that consists of just simple steps like matrix multiplication and thus suitable for embedded hardware like Jetson Nano that has good GPUs for accelerating matrix multiplication. As

a first step towards this goal, we rephrase the optimization (3)-(5) as an unconstrained problem by reformulating the constraints through the Rectified Linear Unit Penalty (ReLU) $\max(0, f)$.

$$\min \mathcal{L}(\mathbf{u}_t), \tag{8}$$

where,

$$\mathcal{L}(\mathbf{u}_t) = c(x_n, y_n) + \sum_t \sum_i w_i^\lambda \lambda_i + \sum_i \sum_t w_i^\mu \mu_i \tag{9}$$

$$\lambda_i = \max(0, \mathbf{g}^i_{bounds}), \mu_i = \max(0, \mathbf{g}^i_{fov}) \tag{10}$$

The subscript i refers to the i^{th} component of the respective constraints. The weights w_i^λ, w_i^μ allows us to trade-off constraint violation and the primary cost.

C. Kinematic Bounds

The constraint function $\mathbf{g}_{bounds}$ consists of bounds on control variables and their derivatives. Its various components are described below

$$\mathbf{g}^1_{bounds} = v_t \leq v_{max}, \quad -v_t \leq v_{min} \tag{11}$$
$$\mathbf{g}^2_{bounds} = \phi_t \leq \phi_{max}, \quad -\phi_t \leq \phi_{min} \tag{12}$$
$$\mathbf{g}^3_{bounds} = \gamma_t \leq \gamma_{max}, \quad -\gamma_t \leq \gamma_{min} \tag{13}$$
$$\mathbf{g}^4_{bounds} = \dot{v}_t \leq \dot{v}_{max}, \quad -\dot{v}_t \leq \dot{v}_{min} \tag{14}$$
$$\mathbf{g}^5_{bounds} = \dot{\phi}_t \leq \dot{\phi}_{max}, \quad -\dot{\phi}_t \leq \dot{\phi}_{min} \tag{15}$$
$$\mathbf{g}^6_{bounds} = \dot{\gamma}_t \leq \dot{\gamma}_{max}, \quad -\dot{\gamma}_t \leq \dot{\gamma}_{min} \tag{16}$$
$$\mathbf{g}^7_{bounds} = \dot{\psi}_t \leq \dot{\psi}_{max}, \quad -\dot{\psi}_t \leq \dot{\psi}_{min} \tag{17}$$

The penalties formed with $\mathbf{g}^i_{bounds}$ are given below

$$\lambda_1 = max(0, v_{min} - v_t) + max(0, v_t - v_{max}) \tag{18}$$
$$\lambda_2 = max(0, \phi_{min} - \phi_t) + max(0, \phi_t - \phi_{max}) \tag{19}$$
$$\lambda_3 = max(0, \gamma_{min} - \gamma_t) + max(0, \gamma_t - \gamma_{max}) \tag{20}$$
$$\lambda_4 = max(0, \dot{v}_{min} - \dot{v}_t) + max(0, \dot{v}_t - \dot{v}_{max}) \tag{21}$$
$$\lambda_5 = max(0, \dot{\phi}_{min} - \dot{\phi}_t) + max(0, \dot{\phi}_t - \dot{\phi}_{max}) \tag{22}$$
$$\lambda_6 = max(0, \dot{\gamma}_{min} - \dot{\gamma}_t) + max(0, \dot{\gamma}_t - \dot{\gamma}_{max}) \tag{23}$$
$$\lambda_7 = max(0, \dot{\psi}_{min} - \dot{\psi}_t) + max(0, \dot{\psi}_t - \dot{\psi}_{max}) \tag{24}$$

We use finite difference $\frac{\mathbf{u}_{t+1} - \mathbf{u}_t}{\Delta t}$ to model the derivatives on the control variables.

D. FOV Constraints

Fig. 1 shows the FOV of a downward-facing camera rigidly attached to the body of FWV. The projection of FOV on the $x-y$ plane can be modeled as the quadrilateral PQRS. We derive the FOV constraints by projecting the FOV prism to $x-z$ and $y-z$ plane. The projection on the $x-z$ plane is shown in Fig.2

Let $(x_t^{target}, y_t^{target})$ be the position of the target vehicle on the $x-y$ plane and the difference between the x-coordinate of target vehicle and FWA $\vec{d_x} = x_t - x_t^{target}$.

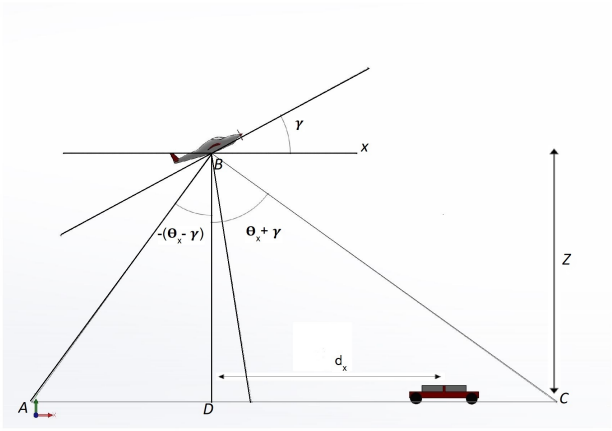

Fig. 2. FOV diagram in x-z plane

For the target be in FOV, we must have

$$\overrightarrow{AD} < \vec{d_x} < \overrightarrow{CD} \tag{25}$$

From Fig.2, we have

$$\overrightarrow{CD} = z_t \tan(\theta_x + \gamma_t) \tag{26}$$
$$\overrightarrow{AD} = z_t \tan(-(\theta_x - \gamma_t)) \tag{27}$$

Therefore, the first half of the FOV constraint can be written as

$$\mathbf{g}^1_{fov} : z_t \tan(-(\theta_x - \gamma_t)) < \vec{d_x} < z_t \tan(\theta_x + \gamma_t) \tag{28}$$

Similarly, by projecting the FOV prism along the $y-z$ plane, we can derive the following the other half of the FOV constraint.

$$\mathbf{g}^2_{fov} : z_t \tan(-(\theta_y - \phi_t)) < \vec{d_y} < z_t \tan(\theta_y + \phi_t) \tag{29}$$

$2\theta_x$ and $2\theta_y$ are the camera's angle of view along x and y axis respectively. The difference between the y-coordinate of target and FWA be $\vec{d_y}$,i.e., $d_y = y_t - y_t^{target}$.

We formulate the following penalties around the FOV constraints.

$$\begin{aligned} \mu_1 = max(0, -z_t \tan(\gamma_t + \theta_x) + \vec{d_x}) \\ + max(0, -z_t \tan(-\gamma_t + \theta_x) - \vec{d_x}) \end{aligned} \tag{30}$$

$$\begin{aligned} \mu_2 = max(0, -z_t \tan(\phi_t + \theta_y) + \vec{d_y}) \\ + max(0, -z_t \tan(-\phi_t + \theta_y) - \vec{d_y}) \end{aligned} \tag{31}$$

E. Soft Constraint on Height

From inequality (28), it is clear that the feasible region of the FOV constraints depends on the height of the FWV above the ground. This relationship is also naturally intuitive because as the FWV gains height, the footprint of its FOV increases. However, this might come at a poor resolution of the images of the ground vehicle. Thus, we introduce a constraint $z_{min} \leq z_t \leq z_{max}$ and the following corresponding penalty

to regulate the height of the FWV. We add (32) to the cost function (9).

$$\mu_h = w_h(max(0, z_{min} - z_t) + max(0, z_t - z_{max})), \forall t. \quad (32)$$

III. Deep Learning inspired Gradient Descent

Optimization (8) represents a non-smooth problem but it has the same form as the common loss functions encountered while training deep neural networks. Thus, state of the art gradient descent variants like ADAM [9] and RmsProp [10] are suitable for solving (8). These variants aim to approximate the curvature of the loss function without computing the Hessian. We briefly summarize the steps in each of these optimizers below.

A. ADAM

Representing, k as the iteration index, the steps of the ADAM optimizer are given by the following equations [9].

$$\mathbf{m}_k = \beta_1 \mathbf{m}_{k-1} + (1 - \beta_1)\nabla\mathcal{L} \quad (33)$$

$$\mathbf{v}_k = \beta_2 \mathbf{v}_{k-1} + (1 - \beta_2)\nabla\mathcal{L}^2 \quad (34)$$

$$\widetilde{\mathbf{m}}_k = \frac{\mathbf{m}_k}{1 - \beta_1^k} \quad (35)$$

$$\widetilde{\mathbf{v}}_k = \frac{\mathbf{v}_k}{1 - \beta_2^k} \quad (36)$$

$$\mathbf{u}_{t+1} = \mathbf{u}_t - \eta \frac{\widetilde{m}_k}{\sqrt{\widetilde{v}_k + \epsilon}} \quad (37)$$

Steps (33)-(34) are the estimates of the mean and variance of the gradient $\nabla\mathcal{L}$ upto iteration $k-1$. The steps (35)-(36) corrects the bias in the mean and variance respectively. The step (37) updates the solution at each iteration. The constant ϵ is added to prevent division by zero.

B. RMSProp

The steps of the RMSProp are given by the following [10], [21].

$$E[\nabla\mathcal{L}^2]_k = 0.9E[\nabla\mathcal{L}^2]_{k-1} + 0.1E[\nabla\mathcal{L}^2]_k \quad (38)$$

$$\mathbf{u}_{t+1} = \mathbf{u}_t - \eta \frac{\nabla\mathcal{L}}{\sqrt{E[\nabla\mathcal{L}_k^2] + \epsilon}} \quad (39)$$

As shown, RMSProp keeps a moving average of the square of the gradient (38) and uses it to update the solution in step (39).

C. Real Time Computation and Warm-Start

Under real-time constraints, it is not possible to solve (8) till convergence. It is common in MPC to run the optimizer for only a fixed number of iterations and execute the obtained control commands. This heuristic is popularly known as the real-time iteration [22]. Another useful heuristic is to warm-start the optimizer at the current iteration with the solutions obtained at the previous iteration. In our context, our warm-start strategy involves storing not only the past solutions but also the vectors $\mathbf{m}$, $\mathbf{v}$ used in ADAM and squared gradients $E[\nabla\mathcal{L}^2]$ used in RMSProp.

IV. Simulation Results

The objective of this section is two-fold. Firstly, to qualitatively analyze the nature of the FWV trajectories for different maneuvers of the ground vehicle. Secondly, to analyze the performance of different gradient descent variants in terms of constraint satisfaction.

Set-up: The following limits on the parameters of the FWA are used in the simulation $z_t \in$ [10, 100] m, $v_t \in$ [4, 20] ms^{-1}, $\dot{v}_t \in$ [0, 10] ms^{-2}, $\gamma_t \in$ [-π/6, π/6] rad, $\phi_t \in$ [-π/5, π/5] rad, $\dot{\gamma}_t \in$ [0, 0.1π] rads^{-1}, $\dot{\phi}_t \in$ [0, 0.1π] rads^{-1}, $\dot{\psi} \in$ [0, 0.5] rads^{-1}, θ_x = 0.26π rad, θ_y = 0.23π rad. The weights associated with the constraint penalties are given below. The same numerical values were used in all the simulation benchmarks and were obtained through trial and error.

$$w_i^{\lambda} = \begin{bmatrix} 1 & 2 & 1 & 0.5 & 0.5 & 0.5 & 0.5 \end{bmatrix}$$
$$w_i^{\mu} = \begin{bmatrix} 5 & 5 \end{bmatrix}, w_h = 100$$

The prediction and control horizon of our MPC was 5 and at each step, only the first control input was executed on the FWV. ADAM and RMSprop optimization techniques were used to solve the optimization problem. The code was implemented in Python3 with gradient computation being performed through Autograd [23] library. Autograd uses the automatic differentiation technique for gradients and thus can handle non-smooth functions as well. The simulation was performed on a laptop with Intel(R) Core(TM) i7-4770 CPU @ 3.40GHz and 16GB RAM. We could perform 25 iterations of ADAM and RMSprop to ensure an average re-planning frequency of 10Hz. We consider two sets of benchmarks depending on whether the target vehicle moves with constant or variable velocity.

A. Constant Velocity Benchmark

Fig.3(a), 3(b) shows the first example in this benchmark where the target vehicle is moving in a straight line with a velocity equal to the maximum velocity of the FWV ($20ms^{-1}$). For such a set-up, one can intuitively imagine that the FWV would be able to track the target vehicle by just increasing its forward speed. The RMSprop optimizer (Fig.3(b)) successfully produces this behavior. The trajectory obtained with ADAM optimizer (Fig.3(a)) is slightly different wherein the change of height is favored in place of acceleration to maximum speed. This can be also reassured by noting that the $x-y$ position of the FWV is behind the target vehicle at the end of the simulation.

Fig.4(a), 4(b) presents the second example in this benchmark where the target vehicle is moving with a velocity lower than the minimum velocity of the FWV. It is clear that the FWV cannot possibly track the target vehicle while moving in a straight line. Thus, we see the FWV execute 3D spiral trajectories around the target vehicle. This behavior agrees with that obtained in the existing state of the art results [13], [14], [15], [16], [17], [18], [19]. However, most of these cited works consider the planar motion while Fig.4(a), 4(b) exposes the full 3D behavior of the FWV.

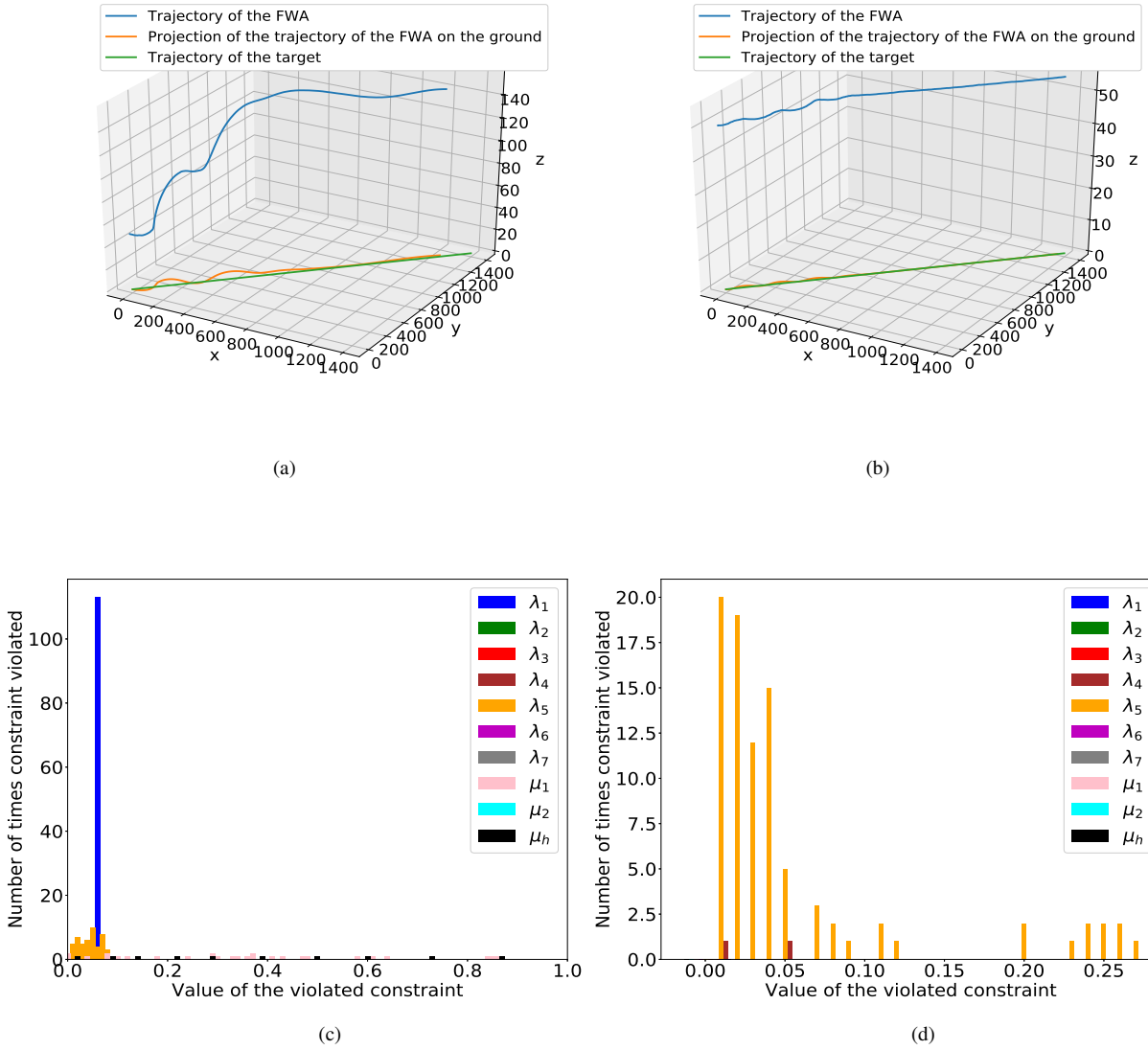

Fig. 3. Fig. 3(a) and 3(b) shows the trajectory generated by ADAM and RMSprop optimizer respectively. Fig. 3(c) and 3(d) shows the constraint violation obtained with ADAM and RMSprop optimizer respectively. The x-axis represents the constraint residuals while the y-axis represents the frequency with which residuals in a particular interval is encountered.

Constraint Violation: Fig.3(c), 3(d), 4(c), 4(d) shows the constraint violation obtained with ADAM and RMSProp optimizer in both the constant velocity examples. The x-axis of these figures represents the numerical value of the constraint residuals while the y-axis represents the number of times a particular residual is encountered during our MPC simulation. As can be seen, constraint residuals remain in the range of $[0, 1]$ for both ADAM and RMSprop optimizer, with the performance of the latter being slightly better. The residual for μ_h (see (32)) is significantly higher than the other penalties. This is because, as mentioned earlier, the restrictions on height were modeled as a soft constraint and this reflected in our tuning of the weights w_i^{λ}, w_i^{μ}, and w_h. Note, it is difficult to correlate the numerical values of these weights with constraint residuals. For example, $w_1^{\lambda} = 1.0$ may lead to lower residual λ_1 than that obtained for μ_h with $w_h = 100$.

B. Variable Velocity Benchmark

In this benchmark, we consider more aggressive maneuvers of the ground vehicle to test the limit of our MPC formulation. In particular, the target vehicle moves with variable velocity and moreover, the changes in its velocity are abrupt.

Fig.5(a), 5(b) shows the example where the target vehicle is employing abrupt de-acceleration at every 100^{th} step of the simulation until it comes to a halt to a complete halt. As can be seen, the RMSProp optimizer leads to a behavior where the FWV moves in almost a straight line at the start. This corresponds to the portion when the target vehicle velocity is greater than the minimum velocity of the FWV. At the end, the FWV has no option but to settle down to a

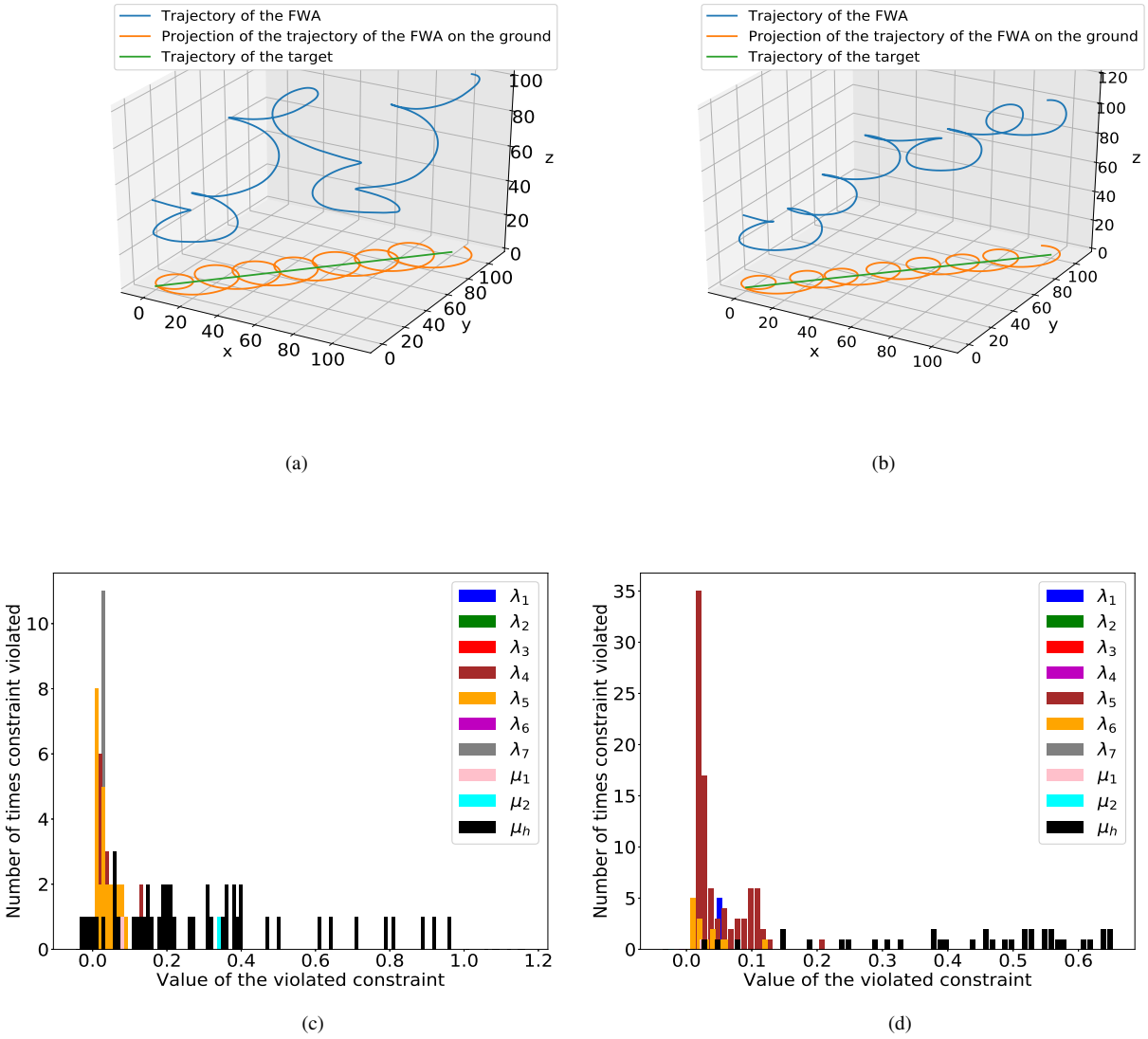

Fig. 4. Fig. 4(a) and 4(b) shows the trajectory generated by ADAM and RMSprop optimizer respectively. Fig. 4(c) and 4(d) shows the constraint violation obtained with ADAM and RMSprop optimizer respectively. The axis notation is similar to Fig.3(c) and 3(d)

spiral trajectory encircling the target vehicle. The trajectory obtained with ADAM optimizer is different at the start but the end portion is similar to that obtained with RMSProp.

Fig.6(a), 6(b) shows the example where the target vehicle starts accelerating from rest till it reaches the maximum velocity of $21ms^{-1}$ which is also the maximum velocity of the FWV. Since at the start, the target vehicle velocity is smaller than the minimum velocity of the FWV, both ADAM and RMSProp lead to 3D spiral trajectories around the target. As the target vehicle velocity increases, the FWV trajectories also becomes simpler.

Constraint Violation: Fig.5(c), 5(d), 6(c), 6(d) summarizes the constraint violation observed in this benchmark. As in constant velocity benchmark, the performance of RMSProp is again better than ADAM. In fact, from Fig.5(c), it can be seen that in the first example, μ_2 has a high residual at many instants during the simulation, indicating that the target vehicle goes out of the FOV on multiple occasions. For the second example, the difference between ADAM and RMSProp in terms of constraint violation is minimal.

V. Conclusions and Future Work

In this paper, we presented a novel MPC formulation for tracking a ground vehicle with a FWV moving in 3D space. A unique aspect of our formulation was to consider a downward-facing camera rigidly connected to the body of the FWV. We formulated a simple yet effective constraint that relates FWV's orientation and its relative position with respect to the target vehicle to the FOV of the camera. In contrast to many existing MPC formulations that use sequential quadratic programming, we made the case of applying gradient descent variants like ADAM and RMSProp

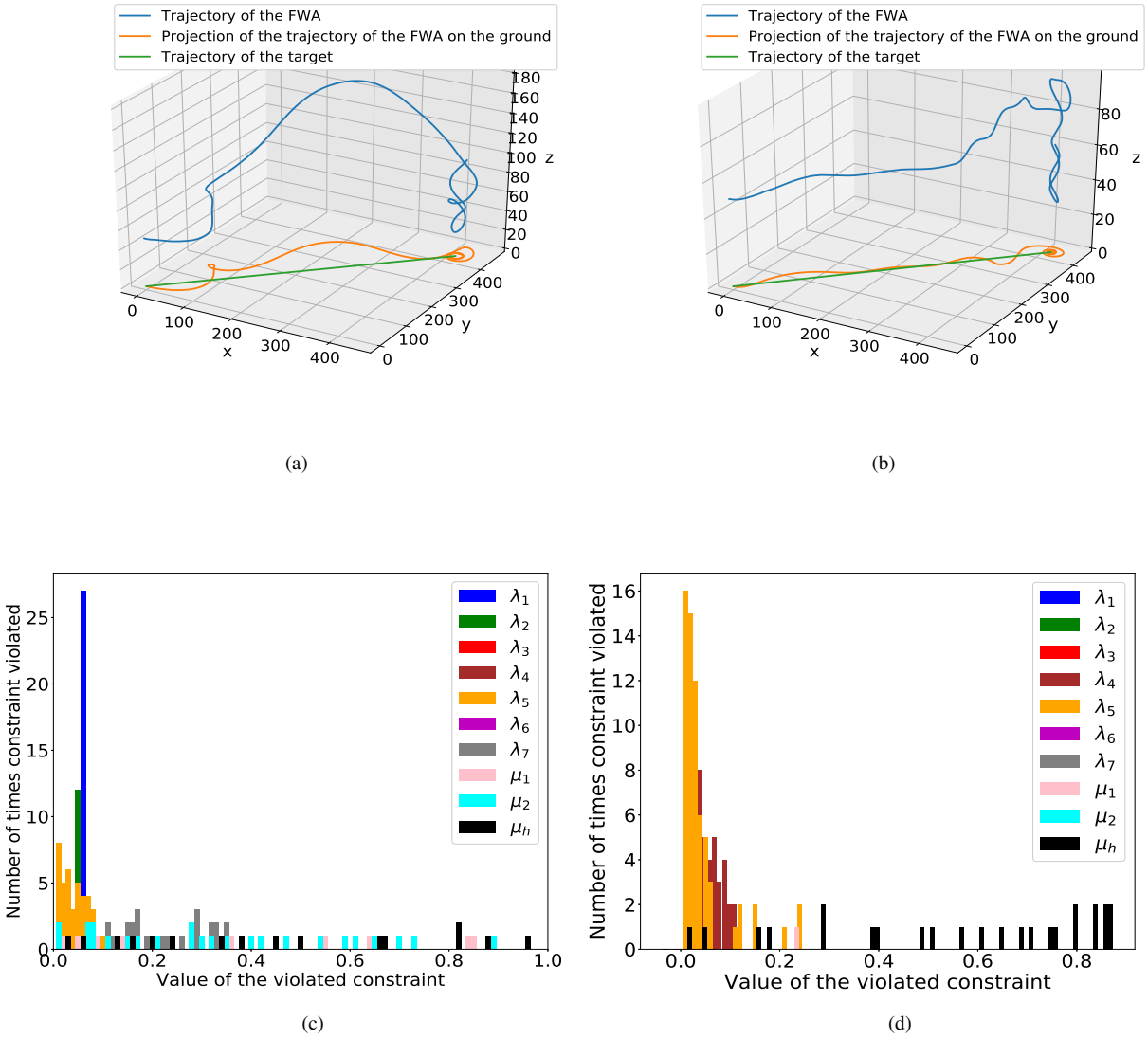

Fig. 5. Fig. 5(a) and 5(b) shows the trajectory generated by ADAM and RMSprop optimizer respectively. Fig. 5(c) and 5(d) shows the constraint violation obtained with ADAM and RMSprop optimizer respectively. The axis notation is similar to Fig.3(c) and 3(d)

to solve the underlying optimization of our MPC. In particular, we showed real-time performance with satisfactory kinematic constraint satisfaction and high-quality tracking. The computationally cheap steps of ADM and RMSProp make our formulation well suited for on-board computation.

In the future, we will incorporate point mass dynamics into our formulation. We are also extending the current work to track multiple vehicles and account for possible occlusions and obstacle avoidance. Experimental validation on a small FWV is also a key part of our future endeavors.

References

[1] R. Wise and R. Rysdyk, "Uav coordination for autonomous target tracking," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, p. 6453.

[2] H. Choi and Y. Kim, "Uav guidance using a monocular-vision sensor for aerial target tracking," *Control Engineering Practice*, vol. 22, pp. 10–19, 2014.

[3] H. Oh, S. Kim, H.-S. Shin, B. A. White, A. Tsourdos, and C. A. Rabbath, "Rendezvous and standoff target tracking guidance using differential geometry," *Journal of Intelligent & Robotic Systems*, vol. 69, no. 1-4, pp. 389–405, 2013.

[4] N. Regina and M. Zanzi, "Uav guidance law for ground-based target trajectory tracking and loitering," in *2011 Aerospace Conference*. IEEE, 2011, pp. 1–9.

[5] H. Chen, K. Chang, and C. S. Agate, "Tracking with uav using tangent-plus-lyapunov vector field guidance," in *2009 12th International Conference on Information Fusion*. IEEE, 2009, pp. 363–372.

[6] P. Theodorakopoulos and S. Lacroix, "A strategy for tracking a ground target with a uav," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 1254–1259.

[7] A. A. Pothen and A. Ratnoo, "Curvature-constrained lyapunov vector field for standoff target tracking," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 10, pp. 2729–2736, 2017.

[8] M. Nieuwenhuisen and S. Behnke, "Search-based 3d planning and trajectory optimization for safe micro aerial vehicle flight under sensor visibility constraints," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9123–9129.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization.

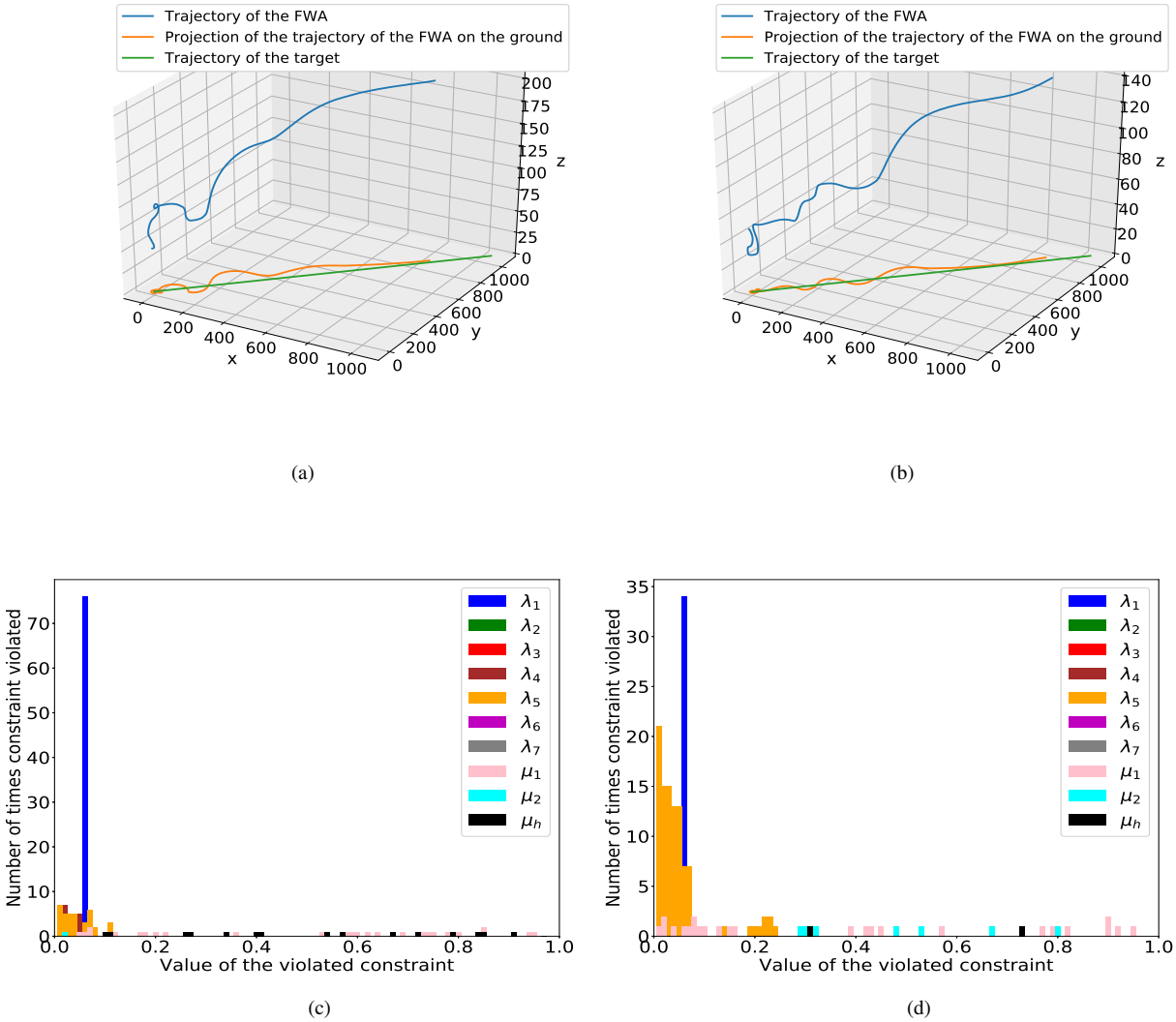

Fig. 6. Fig. 6(a) and 6(b) shows the trajectory generated by ADAM and RMSprop optimizer respectively. Fig. 6(c) and 6(d) shows the constraint violation obtained with ADAM and RMSprop optimizer respectively. The axis notation is similar to Fig.3(c) and 3(d)

iclr (2015)," *arXiv preprint arXiv:1412.6980*, vol. 9, 2015.

[10] Y. Bengio, "Rmsprop and equilibrated adaptive learning rates for nonconvex optimization," *corr abs/1502.04390*, 2015.

[11] M. Brandao, K. Hashimoto, and A. Takanishi, "Sgd for robot motion? the effectiveness of stochastic optimization on a new benchmark for biped locomotion tasks," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 39–46.

[12] B. O'Donoghue, G. Stathopoulos, and S. Boyd, "A splitting method for optimal control," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2432–2442, 2013.

[13] J.-P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans, "Urban target search and tracking using a uav and unattended ground sensors," in *2015 American Control Conference (ACC)*. IEEE, 2015, pp. 2401–2407.

[14] J. Kim and J. L. Crassidis, "Uav path planning for maximum visibility of ground targets in an urban area," in *2010 13th International Conference on Information Fusion*. IEEE, 2010, pp. 1–7.

[15] Y. Watanabe and P. Fabiani, "Optimal guidance design for uav visual target tracking in an urban environment," *IFAC Proceedings Volumes*, vol. 43, no. 15, pp. 69–74, 2010.

[16] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous uav surveillance in complex urban environments," in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2. IEEE, 2009, pp. 82–85.

[17] J. Wu, H. Wang, N. Li, P. Yao, Y. Huang, and H. Yang, "Path planning for solar-powered uav in urban environment," *Neurocomputing*, vol. 275, pp. 2055–2065, 2018.

[18] X. Zhao, F. Pu, Z. Wang, H. Chen, and Z. Xu, "Detection, tracking, and geolocation of moving vehicle from uav using monocular camera," *IEEE Access*, vol. 7, pp. 101 160–101 170, 2019.

[19] P. Theodorakopoulos and S. Lacroix, "Uav target tracking using an adversarial iterative prediction," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 2866–2871.

[20] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, *et al.*, "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature methods*, pp. 1–12, 2020.

[21] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[22] M. Diehl, H. G. Bock, and J. P. Schlöder, "A real-time iteration scheme for nonlinear optimization in optimal feedback control," *SIAM Journal on control and optimization*, vol. 43, no. 5, pp. 1714–1736, 2005.

[23] D. Maclaurin, D. Duvenaud, and R. P. Adams, "Autograd: Effortless gradients in numpy," in *ICML 2015 AutoML Workshop*, vol. 238, 2015.