Perception Action Aware-Based Autonomous Drone Race in a Photorealistic Environment

MUHAMMAD KAZIM1, (Member, IEEE), ADEEL ZAIDI1, (Member, IEEE), SHAFQAT ALI2, MUHAMMAD TASKEEN RAZA3, GHULAM ABBAS4, (Senior Member, IEEE), NASIM ULLAH5, (Member, IEEE), AND AHMAD AZIZ AL-AHMADI5, (Member, IEEE)

1Research Institute of Intelligent Control and Systems, Harbin Institute of Technology, Harbin 150001, China
2Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, China
3Department of Electrical Engineering, Lahore College for Women University, Lahore 54000, Pakistan
4Department of Electrical Engineering, The University of Lahore, Lahore 54000, Pakistan
5Department of Electrical Engineering, College of Engineering, Taif University, Taif 21944, Saudi Arabia

Corresponding author: Ghulam Abbas (ghulam.abbas@ec.uol.edu.pk)

This work was supported by Taif University, Taif, Saudi Arabia, under Taif University Researchers Supporting Project (TURSP-2020/121).

ABSTRACT The development of autonomous, fast, agile small Unmanned Aerial Vehicles (UAVs) brings up fundamental challenges in dynamic environments with fast and agile maneuvers, unreliable state estimation, imperfect sensing, coupling action, and perception in real-time under severe resource constraints. However, autonomous drone racing is a challenging research problem at the intersection of computer vision, planning, state estimation, and control. To bridge this, we propose an approach in the context of autonomous, perception-action aware vision-based drone racing in a photorealistic environment. Our approach integrates a deep convolutional neural network (CNN) with state-of-the-art path planning, state estimation, and control algorithms. The developed deep learning method is based on computer vision approaches to detecting the gates and estimating the flyable area. The planner and controller then use this information to generate a short, minimum-snap trajectory segment and send corresponding motor commands to reach the desired goal. A thorough evaluation of our proposed methodology has been carried out using the Gazebo and FlightGoggles (photorealistic sensor) environments. Extensive experiments demonstrate that the proposed approach outperforms state-of-the-art methods and flies the drone more consistently than many human pilots. Moreover, we demonstrated that our proposed system successfully guided the drone through tight race courses, reaching speeds up to 7m/s of the 2019 AlphaPilot Challenge.

INDEX TERMS Control, computer vision, drone race, deep learning, path planning, photo-realistic environment, state estimation.

I. INTRODUCTION

A. MOTIVATION

In recent years, autonomous Unmanned Aerial Vehicles (UAVs) or drones in diverse scenarios have gained much attention. UAVs are attracting increased interest across various communities such as defense, emergency response, disaster relief, healthcare, agriculture, mining, infrastructure development, sports, education, and many other areas [1]. One of the challenging tasks for UAVs is competitive drone racing. Each UAV is controlled by a human pilot in a drone racing, who receives a first-person-view live stream from an onboard camera and flies the drone by sending commands via a radio transmitter. Human drone pilots need years of training to master advanced navigation but frequently involve crashes. However, the maneuverability and speed of these human pilots still outperform the existing control algorithms for autonomous UAVs. The performance of autonomous racing drones is still far from human pilots’ speed, versatility, and robustness; hence, a lot of research and innovation is needed to fill this gap. Developing fully autonomous racing UAVs to achieve optimal performance is arduous due to dynamic modeling, onboard perception, trajectory generation, localization, and mapping. Due to the miniaturization of onboard electronics and technological advancement, we expect that fully autonomous racing drones will soon compete against human pilot experts very effectively and efficiently.

In 2019, Lockheed Martin and the Drone Racing League launched the AlphaPilot Challenge [2], [3], an open innovation challenge with a grand prize of $1 million, in order to get...
the AI-enabled autonomous drone race. Current autonomous drones have very little onboard decision-making; they will almost always follow particular human commands and rarely accomplish higher-level tasks. Artificial Intelligence algorithms are mainly developed only for simulations, and from simulation to the real world, flying is a big challenge. Another challenge of autonomous drone flight is the perception based on cameras because the faster the drone maneuver, the more blurred the image gets.

This research work’s main motivation and goal are to develop fully autonomous UAVs that navigate through a racecourse using AI, computer vision, and control theory. This research work and Alphapilot challenge aim to beat one day the best human pilot by pushing the limits in terms of speed and course size to advance state of the art, while other autonomous drone races [4], [5] focus on complex navigation. Because of its high speed, autonomous drone racing raises fundamental challenges in real-time state estimation, perception, planning, and control, to beat the best human pilot. The limited computational power and complex visual environments (e.g., motion blur and low light) are challenging tasks for autonomous drone racing.

This paper is organized as follows. Section I presents the introduction, in which we discuss the motivation behind this research work, the latest related work in this area, and our contribution. Section II provides the drone race format, mathematical modeling of UAV, drone model, and classification. The complete methodology is presented in section III, which comprises gate detection (perception system), control system, and drone interface. Section IV presents the performance of the experiments in simulated and real scenarios, with their results, and finally, section V concludes the paper and points towards future research directions.

### B. RELATED WORK

This section puts our proposed methodology into context, focusing on the most related work. The cumulative complexity of the subproblems, i.e., object detection and recognition [6], non-linear control [1], and path planning [7] makes drone racing an exciting and challenging problem. The rapid development in AI has contributed to broader use in the robotics community of many novel technologies, such as in deep learning [8] and reinforcement learning [9].

With the recent work in AI and the progress in powerful computer chip development, deep convolutional neural networks (CNNs) became the standard approach for AI and computer vision applications [10], [11].

In 2013, Ross et al., in their paper [12] presented Learning monocular reactive UAV control in cluttered natural environments, in which the task is the collision-free flight in the forest. The network input was imaging from a forward-facing camera, network output was desired lateral speed, and the training methodology was supervised learning with recorded data from a human pilot. After initial training on the expert data, the policy is refined using datasets Aggregation. However, the limitation in this paper is that the vehicle dynamics are not taken into account, 2D sideways motion only (roll left or right), and the constant linear speed limit is less than 1.5m/s. In 2017, in Smolyanskiy et al., the images were captured from a forward-facing camera following the forest trail, and the heading angles were provided to the network input. The network output was the lateral offset to the forest trail was the network output, the training methodology was supervised learning on hand-recorded data, and the limitations were 2D unicycle motion, vehicle dynamic not taken into account, and constant linear speed was less than 2m/s. In 2018, Jung et al., in their paper [8], in which the task is to navigate through a set of gates. The image from the forward-facing camera was network input and segmentation of gate, and the velocity commands are computed to align optical center with gate center, and network inference is performed onboard was network output. However, their limitations are 2D unicycle motion, and vehicle dynamics are not considered. In 2019, Muller et al., [13], demonstrated that the task of a UAV is to navigate through a set of gates as fast as possible. The network input was an image from a forward-facing camera and platform state, and low-level control commands (body rates and thrust) were network output. The network is trained using an ensemble of classical controllers; each classical controller is evaluated, and the best one is chosen to imitate, but the limitations are that they assume perfect knowledge of the system state and only work in simulation.

Despite these progresses in AI and control, the optimal output representation for learning-based algorithms that couple perception and control is an open question. This paper presents a complete system for solving the missions designed in the AlphaPilot drone racing 2020 competition in a fully AI-enabled autonomous manner and developing generic and versatile approaches that can be utilized in many other applications.

### C. CONTRIBUTION

The combination of robotics and AI will change the future, and AI-enabled autonomous technologies will help researchers, scientists, astronauts, military service, and first responders demand dangerous jobs more safely and efficiently. The main contribution of this work is given as follows:

- The proposed hybrid approach systems based on machine learning (for perception) and models (for control) are more robust than systems based exclusively on one of the two. It also does not require any explicit map of the environment and runs fully onboard.
- Several computer vision methods were used to improve the network’s ability to generalize and increase the training dataset and image shifting, flipping, wrapping, and color manipulation methods.
- Two approaches were considered for detecting the gate and estimating the flyable area in real-time. The first approach was a detection and classification network like...
A. DRONE RACE FORMAT

We designed a complete track system with gates at different distances from the ground in the Gazebo environment, as shown in Fig. 1, in which a UAV can fly and navigate in a simple 3D track marked with squared gates (dimensions are 0.5m × 0.5m), the UAV enters with a similar pose to complete a full lap at best speed without collisions with gates. The squared, or parallelogram gates are marked with unique visual fiducial markers (ARuco tags with 4 dictionary) that can help to estimate the pose and size of the gate, as given in the attached video.1

The drone race course format was provided by the AlphaPilot team [2], [15], in which the UAV starting location is in the upper right corner as shown in Fig. 2. In this challenge, the UAV must traverse gates in order along the course marked (Gate 10, 21, 2, 13, 9, 14, 1, 22, 15, 23, 6) in red and must finish by passing through the gate in the lower-left corner of the figure (gate 6). In the figure below, gate IDs belonging to gates along the race path are marked in yellow, while the gate IDs in blue are not part of the racecourse.

1https://youtu.be/TiL02fl52JU

B. MATHEMATICAL MODELING OF UAV

In the case of the autonomous drone race, an effective dynamic model of the system is needed to design a controller that captures the non-linearities of UAV flight dynamics. In this paper, X-configuration-based six DOF underactuated UAV (Intel aero RTF UAV [16]) is considered. Six DOF includes three translational motions and three rotational motions around three axes. The schematic configuration of the UAV consists of four propellers which are divided into two pairs, Clockwise (CW) and Counterclockwise (CCW), that are mounted on four motors in two orthogonal directions to compensate for the effect of reactive torque. The front F1 and rear F3 Propellers rotate CCW, while the lateral propellers (F2 and F4) rotate clockwise direction to produce the lift force and balance the yaw’s torque required. The roll rotation is achieved by changing the speed of the F2 and F4 motors of the propeller. The pitch rotation is realized by adjusting the speed of the F1 and F4 motors of the propeller.

To determine the model dynamics of UAV, some assumptions are established as follows:

Assumption 1: The structure of UAV is symmetrical.

Assumption 2: The body frame of UAV and propellers has a rigid body with six DOF that can be derived by using Newton-Euler equations.

Assumption 3: The drag and thrust forces are equal to the size of the square of the propeller’s size.

1) THE REFERENCE FRAMES AND TRANSFORMATION

The mathematical model of the UAV is derived by using (1), (2) and (3) assumptions. Let \( E = \{O_e, X_e, Y_e, Z_e\} \) represents an inertial-fixed frame and \( B = \{O_b, X_b, Y_b, Z_b\} \) represents a body-fixed frame whose origin \( O \) is at the center of UAV mass. The orthogonal rotation matrix is \( R_B^E \in SO(3) \) with a determinant \( \text{det}(R_B^E) = 1 \), where \( (R_B^E)^{-1} = (R_B^E)^T \) and can be expressed in (1), as shown at the bottom of the next page, where \( \phi, \theta, \psi \) represent the roll, pitch and yaw angles respectively. The position of the UAV in the inertial or earth-fixed frame is indicated by \( \xi = [x, y, z]^T \), the successive rotations of the yaw \( \psi \), pitch \( \theta \) and roll \( \phi \) attitude or by obtaining Euler angles \( \eta = [\phi, \theta, \psi]^T \). The translational velocities are expressed by \( \nu = [v_x, v_y, v_z]^T \) in the \( x \), \( y \) and \( z \) directions. The angular velocities in body-fixed frame are \( \omega = [p, q, r]^T \), where \( p(t) \) is the x-axis roll rate, \( q(t) \) is the y-axis pitch.
rate, and \( r(t) \) is the z-axis yaw rate. The \((X_b, Y_b, Z_b)\) transformation from the body-fixed coordinates to the earth-fixed coordinates \((X_e, Y_e, Z_e)\) is given by \( R_E^b = (R_E^b)^T \). The UAV kinematic equations \([17], [18]\) can be expressed as:

\[
\dot{\mathbf{p}} = \mathbf{v}
\]

(2)

If \( \forall t \) the pitch angle \( \theta(t) \in (-\pi/2, \pi/2) \), roll angle \( \phi(t) \in (-\pi/2, \pi/2) \), and yaw angle \( \psi(t) \in (-\pi, \pi) \), then the derivatives of Euler angle can be related to the body-fixed angular velocities by:

\[
\dot{\eta} = \mathbf{W}(\eta) \omega
\]

(3)

and \( \det(\mathbf{W}) = \sec(\theta) \). Therefore, if the pitch angle satisfies \( \theta \neq (2k - 1)\pi/2, (k \in \mathbb{Z}) \), the matrix \( \mathbf{W}(\eta) \) is invertible.

2) FORCE EQUATIONS AND PROPELLER DYNAMICS

The combination of mass and inertial acceleration, using the Newton-Euler equation, is equal to all the forces exerted on the UAV, i.e., total thrust \( T \), weight \( mg \), and the drag \( F_w \):

\[
m \ddot{x} = -m g e_z + T R_E^b e_z + F_w
\]

(4)

where \( m \) is the UAV mass, \( g \) is the gravitational acceleration, \( e_z = [0, 0, 1]^T \) is the vector unit and \( T = \sum_{i=1}^{4} f_i = b \sum_{i=1}^{4} \Omega_i^2 \) is the maximum thrust in the (-z) direction generated by four propellers. The reaction torque caused by air drag is \( Q_i = k \Omega_i^2 \), where \( \Omega_i \) represents the speed of the propeller \( i, k > 0 \) is the thrust constant and \( b > 0 \) is the drag factor on the propeller.

3) MOMENT EQUATIONS

The time derivatives of the UAV \((H = I\omega)\) angular momentum are equal to the external moments \( \tau_{total} \), on the system including gyroscopic moments \( \Gamma \), and the torques \( \tau \). The moment equations are therefore given by:

\[
\frac{d}{dt} H = \dot{H} + \omega \times H = \tau_{total}
\]

(5)

\[
I \dot{\omega} + \omega \times (I \omega) = \tau + \Gamma
\]

(6)

The total inertial matrix \( I_r \in R^{3 \times 3} \) under assumption 3 is a positive definite symmetric matrix expressed in the body-fixed frame. The vector \( \Gamma \) corresponds to the gyroscopic torque is given as:

\[
\Gamma = \sum_{i=1}^{4} I_r (\omega \times z_e)(-1)^{i+1} \Omega_i
\]

(7)

where \( I_r \) is the motor axis moment of inertia and \( \tau = [\tau_\phi, \tau_\theta, \tau_\psi] \), is the external propeller torques.

Let \( d \) is the distance between the UAV center of mass and the center of propeller. The control torques produced by the four propellers are then given by:

\[
\tau = \begin{bmatrix}
\tau_\phi \\
\tau_\theta \\
\tau_\psi
\end{bmatrix} = \begin{bmatrix}
bd (\Omega_2^3 - \Omega_1^3) \\
bd (\Omega_3^3 - \Omega_1^3) \\
K(\Omega_2^3 + \Omega_3^3 - \Omega_1^3 - \Omega_4^3)
\end{bmatrix}
\]

(8)

4) MOTOR MIXING

To simplify the computation of the real control inputs, i.e., \( \omega_i(i = 1, 2, 3, 4) \), the equations of force and torque are computed as follow:

\[
\begin{bmatrix}
T \\
\tau_\phi \\
\tau_\theta \\
\tau_\psi
\end{bmatrix} = \begin{bmatrix}
b & b & b & b \\
0 & -bd & 0 & bd \\
-bd & 0 & bd & 0 \\
-k & k & -k & k
\end{bmatrix} \begin{bmatrix}
\Omega_1^3 \\
\Omega_2^3 \\
\Omega_3^3 \\
\Omega_4^3
\end{bmatrix} \approx M \hat{\Omega}
\]

(9)

Since the controller determines the control inputs \((T, \tau_\phi, \tau_\theta, \tau_\psi)\), these inputs are converted by inverting the above equation into the desired angular velocities of the propellers. Typically, this transformation is called motor mixing.

5) UAV DYNAMIC EQUATION

Because of the aerodynamic uncertainty, the UAVs motion equations are arranged together and rewritten as follows:

\[
\begin{aligned}
\dot{\mathbf{p}} &= \mathbf{v} \\
\dot{\eta} &= \mathbf{W}(\eta) \omega \\
m \ddot{x} &= -m g e_z + T R_E^b e_z + F_w \\
I \dot{\omega} + \omega \times (I \omega) &= \tau + \Gamma.
\end{aligned}
\]

(10)

After putting the values and rearranging the above equations become:

\[
\begin{aligned}
\dot{\mathbf{p}} &= \mathbf{v} \\
\dot{\eta} &= \mathbf{W}(\eta) \omega \\
m \ddot{x} &= -m g e_z + m^{-1} T R_E^b e_z + m^{-1} F_w \\
I \dot{\omega} + \omega \times (I \omega) &= \tau + \Gamma
\end{aligned}
\]

(11)

C. SIMULATOR SPECIFICATIONS

Using a simulation environment allows researchers and practitioners to isolate a single task from a complex pipeline without performing experiments using a costly physical quadrotor platform. FlightGoggles simulator \([5]\) provides a stereo camera at 60 FPS, 0.32m baseline, and the changeable resolution of 1024 \times 768 pixels for default. The camera pair has zero distortion and a 70-degree Field of View. It also has a high rate at 960 Hz IMU sensor with an accelerometer variance of 0.005 and gyroscope variance of 0.003. In addition, ground
III. METHODOLOGY

The methodology comprises six main subsystems: sensor interface, perception, state estimation, path planning, and control pipeline to address the fast, agile, and robust flight of quadrotors in the drone racing arena. The quadrotor’s sensor interface pipeline is equipped with cameras, IMU, and other sensors. The information from the sensor interface is then used in the perception module, which mostly does two things. First, a gate detection component uses computer vision and deep learning to detect the gates and predict the goal direction in local image coordinates from the forward-facing camera. The VINS-Mono provides us with information about the motion of the quadrotor. This information is then used by the state estimation module, which uses the information about the gate corners and the odometry data to compute a consistent global map of the gate layout and the state of the quadrotor. We know where our quadrotor is, how it is oriented in the world and where it is relative to the gates through this. The path planning module then takes this information and generates a minimum snap trajectory, using the minimum snap trajectory generation method [19]. This is then output in the form of a reference trajectory fed to the control component of the pipeline. In a control system, the reference trajectory is used to compute the actual commands of the rotors of the quadrotor platform to follow the trajectory as accurately as possible. The computed commands are then passed to the drone interface, and the complete system architecture is shown in Fig. 3.

The following section discusses the methodology of our proposed perception, state estimation, path planning, and control systems.

A. PERCEPTION SYSTEM

We proposed and developed a deep learning approach for detecting the gates and estimating the flyable area in the perception system. The perception pipeline is divided into two parts, gate detection and Visual Inertial Odometry (VIO); for gate detection, we built a modular algorithm consisting of three pipelines: preprocessing, detection, and post-processing. This agile architecture enabled us to rapidly test different methods for detection and implement the fastest and most accurate one.

1) PIPELINE 1

This is the preprocessing pipeline in which we have curated the dataset list and filtered the lists with missing polygons. The biggest challenge was the dataset annotation. Polygon annotation is a precise way to annotate objects by selecting a series of x and y coordinates along its edges. The first polygon annotation was used to identify and train the flyable area features of gates, while the second polygon annotation was used to identify and train the classifier with gates features. Using the provided inner polygons of the gate frame alone was insufficient to train a robust classifier and generalize to unseen frames during the race. Therefore we added another polygon annotation to the dataset that helped us identify and train the classifier with gates features rather than the flyable area feature. Our theory is that if we could locate the gate, we could find the flyable area easily using a post-processing pipeline. Consequently, we created masks for training.

The preprocessing pipeline also included a camera calibration method that used the provided gate images to remove the deformation of the images caused by the lens used for taking the pictures for the data set. Several data augmentation methods were introduced to improve the network’s ability to generalize and increase the training dataset. The augmentation included image shifting, flipping, wrapping, and color manipulation methods. The pseudocode of image augmentation is provided in algorithm 1.

Algorithm 1: Image Augmentation

1) Calibrate the camera:
   return calibratecamera (objpoints, imgpoints)
2) Undistort the image:
   return undist(img, mtx, dist)
3) Convert to grayscale:
   return gray(undist)
4) Gaussian filter:
   return img_blur(gray)
5) Shape and Corners Detection:
   return shape, corners
6) Define the four Source Points:
   return src
7) Define the four destination Points:
   return dst
8) Transform matrix:
   return M(src, dst)
9) Warp images to a top-down view:
   return warpPerspective(undist, M, img_size)

2) PIPELINE 2

Two approaches were considered for detecting the gate and estimating the flyable area in real time. The first approach was
a detection and classification network like the YoloV3 [14]. The second approach was to use a pixel-wise segmentation network. We decided to use a real-time pixel-wise segmentation to improve the flyable area’s accuracy. We trained, tested and evaluated four segmentation networks ENet network [20] and DeepLabv3 [21] with xception and resnet backends, and Unet [22]. The DeepLabV3 network was chosen for the final pipeline based on the gate segmentation accuracy and speed. DeepLabV3 uses dilated convolution, Multi-Grid, and Atrous Spatial Pyramid Pooling (ASPP) [21]. We trained the network using the resnet backend using a GTX 2080TI GPU.

3) PIPELINE 3
After the network was trained in an end-to-end fashion, the predictions were thresholded to create a binary mask. This post-processing algorithm consisted of several computer vision methods to filter and estimate the flyable area. The second step in the algorithm is to detect the outer and inner corners of the predicted gate and return their pixel coordinates.

4) DATASET COLLECTION AND DISTRIBUTION
The network is trained on a training dataset contains roughly 9,300 images, a total of 2.8GB, and the ground truth labels for this dataset are available here. All the images are taken with Canon DSLR with an 18mm lens, and the square sizes of the images are 19 × 19mm. While the test dataset contains roughly 1,000 images totalling 360MB. This test dataset should be used to see how well the algorithms perform on unseen images.

B. OPTIMIZATION BASED VIO METHOD
The optimization-based VIO method, the VINS-Mono [23] was chosen due to its high accuracy and robustness under high-speed motion. VINS-mono is a graph-optimization-based solution to a non-linear visual-inertial system. The VINS-Mono utilized the pre-integration of IMU jointly with feature observations from cameras and the pre-integration of measurements from IMU. The below equation is the objective function of VINS-Mono, which features a tightly coupled and sliding window-based formulation.

\[
X^* = \arg \min \{ \| \rho_p - H_p X \|^2 + \sum_{k \in B} \| \rho_p(y_{k_{t+1}}, X) - \rho_p(y_{k_{t+1}}) \|^2 + \sum_{(l,j) \in C} \rho(y_{l}(z_{l}^c, X) - \rho_l(z_{l}^c)) \} \tag{12}
\]

For brevity, we have omitted the explanation of (12) here and refer the reader to [23] for detailed definition and derivation of the variables and symbols in the function. Equation (12) is linearized and solved iteratively using Ceres [24], an incremental linear solver. The key to the robustness of VINS-Mono lies in its on-the-fly estimator initialization and loop closure detection. The working pipeline of VINS-Mono is given in Fig. 4.

C. STATE ESTIMATION
The perception, i.e., gate detection (gate corners) and VINS-Mono (odometry data) are then used by the state estimation module of the system. Here we used Extended Kalman Filter (EKF) to fuse the information about the gate corners and the odometry data to compute a consistent global map of the gate layout and the state of the quadrotor. So we know where our drone is, how it is oriented in the world and where it is relative to the gates. The EKF is the extension to the nonlinear case, where we use the Jacobian matrix of the transition \( G_t \) and observation functions to compute a point-wise linear estimate and then do the updates. We define the Extended Kalman Filter (EKF) algorithm 2 by following [25]. We refactor it to include separate Predict and Update methods and to use our notations, \( u_t \) is the control input, \( x_t \) is the states and \( \bar{\mu}_t \)

Algorithm 2: EKF Algorithm

Function PREDICT(\( \mu_{t-1}, \Sigma_{t-1}, u_t, \Delta t \))

\[
\bar{\mu}_t = g(u_t, \mu_{t-1})
\]

\[
G_t = g'(u_t, x_t, \Delta t)
\]

\[
\Sigma_t = G_t \Sigma_{t-1} G_t^T + Q_t
\]

return \( \bar{\mu}_t, \Sigma_t \)

Function UPDATE(\( \bar{\mu}_t, \Sigma_t, z_t \))

\[
H_t = h'(\bar{\mu}_t)
\]

\[
K_t = \Sigma_t H_t^T (H_t \Sigma_t H_t^T + R_t)^{-1}
\]

\[
\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))
\]

\[
\Sigma_t = (I - K_t H_t) \Sigma_t
\]

return \( \mu_t, \Sigma_t \)

Function EXTENDEDKALMANFILTER

\( u_t = \text{COMPUTECONTROL(} \mu_{t-1}, \Sigma_{t-1} \}\)

\( \bar{\mu}_t, \Sigma_t = \text{PREDICT(} \mu_{t-1}, \Sigma_{t-1}, u_t, \Delta t \) \)

\( z_t = \text{READSENSOR()} \)

\( \mu_t, \Sigma_t = \text{UPDATE(} \bar{\mu}_t, \Sigma_t, z_t \) \)

\[\text{FIGURE 4. Working pipeline of VINS-Mono.}\]
and $\Sigma_t$ are the mean and covariance while $Q_t$ are the process noise.

### D. PATH PLANNING

The controller is designed to deal with the deviations from the desired path and disturbances that occur during path execution, and we need to guarantee that the planned trajectory is feasible for the drone to follow and does not diverges from the desired path. Minimum snap trajectory generation [26], the fourth derivative of position (angular acceleration), is used to compute a global reference trajectory that passes through all track gates. We need to optimize the trajectory for which the cost of the fourth derivative of the trajectory can be expressed as,

$$J = \int \|x^{(4)}\|^2 dt$$

(13)

The path can be represented as a piecewise polynomial between the waypoints, and the condition that minimizes the cost is $x^{(4)} = 0$. Thus in practice a 7th order polynomial is used to generate the path.

$$x_n(t) = C_n + C_n(t - t_n) + C_n(t - t_n)^2 + \ldots + C_n(t - t_n)^7$$

(14)

where $n$ corresponds to the index of the spline segment containing $t$, and $t_n$ is the time associated with the start of segment $n$.

To optimize snap of a segment, we minimize the integral from 0 to $d_n$ of the 4th derivative of the polynomial, where $d_n$ is the duration of segment $n$, equal to $t_{n+1} - t_n$. The cost of the fourth derivative of the trajectory can be expressed as, in one dimension, $x$

$$s_n = \int_0^{d_n} \left[ \frac{d^4}{dt^4} x_n(t) \right]^2 dt.$$  

(15)

Thus, the total snap $s$ over a given dimension is:

$$s = \sum_{n=0}^{N} s_n$$

(16)

where $N$ is the total number of segments on the trajectory.

### E. CONTROL SYSTEM

A cascaded PID controller is implemented for appropriate thrust and angular velocities in the built-in low-level angular speed controller. The outer loop of the controller aims to control the position and yaw of the drone along a given trajectory, while the inner loop of the controller achieves pursuit of the desired altitude and attitude. The dynamics of the drone presented in (11) are simplified, assuming that the provided low-level controller perfectly regulates the angular velocities. The proposed control strategy consisted of Altitude controllers: $k_{p-z}, k_{d-z}, k_{i-z}$.

$$u_1 = k_{p-z}(z_t - z_a) + k_{d}(\dot{z}_t - \dot{z}_a) + \ddot{z}_t + \dddot{z}_n + k_i e$$

(17)

where $\dddot{z}_n$ is the feed-forward component of the controller. Lateral position controller: $k_{p-x}, k_{d-x}, k_{p-y}, k_{d-y}$,

$$\dot{x} = k_{p-x}(x_t - x_a) + k_{d-x}(\dot{x}_t - \dot{x}_a).$$

(18)

Roll-Pitch controller: $k_{p-roll}, k_{p-pitch}$.

$$\begin{bmatrix} p_c \\ q_c \\ r_c \end{bmatrix} = \begin{bmatrix} R_{21} & -R_{11} \\ R_{22} & -R_{12} \end{bmatrix} \times \begin{bmatrix} k_{p-roll}(\frac{z - \dot{z}}{a_x}) \\ k_{p-pitch}(\frac{z - \dot{z}}{a_y}) \end{bmatrix}$$

(19)

where indexed ‘‘R’’ are the corresponding rotation matrix as given in (1), and $c = \frac{F}{m}$, in which “F” is the lifting force. A ‘‘P’’ controller is used to control the UAV’s yaw

$$r_c = K_{p-yaw}(\psi_t - \psi_a)$$

(20)

and Body rate controller are:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} k_{p-roll}(p_t - p_a) \\ k_{p-q}(q_t - q_a) \\ k_{p-yaw}(r_t - r_a) \end{bmatrix}$$

(21)

which takes the moments of inertial and body rates to calculate the roll-pitch controller, for both simulation and hardware purposes.

### IV. RESULTS AND DISCUSSION

#### A. GATE DETECTION: PERCEPTION SYSTEM

When designing the perception and computer vision test, rapidity, computational efficiency, perception, and robustness have been our core objectives. There were around 9000 images for training the gate detection with poor labeling. So, we divided the gate detection part into three sections, preprocessing, detection and postprocessing, as discussed in section III.A. We have curated the dataset list in preprocessing pipeline and filtered the lists with missing polygons. Consequently, we created masks for training. Sample masks annotation and augmented images are shown in Fig. 5.

DeepLabV3 with exception and reset backend provides the more accurate results, as shown in Fig. 6 as compared to others. The final accuracy of our masks was 95.6% mIoU with our validation dataset and 96.1% mIoU on our manually annotated images.

During gate detection, another technical challenge was generated with the occlusion, as shown in Fig. 7 and multiples annotated images.

B. OPTIMIZATION BASED VIO METHOD

We applied a state-of-art feature-based and non-linear optimization-based VINS-Mono algorithm to get the information about the drone’s motion in the FlightGoggle simulator, as shown in Fig. 9 which is photorealistic and...
which reflects UAV’s real dynamics. VINS-Mono also provides online temporal calibration, which solves the synchronization problem of sensors. The Root Mean Square Error (RMSE) between the ground truth and estimated states for VINS-Mono is around 1.7215m, while the average FPS is 14.9. As VINS-Mono solves non-linear optimization to estimate states, it uses memory the most, but it is accurate. It provides online temporal calibration so that poor calibration can be ignored and hardware synchronization is unnecessary. In addition, it supports loop-closure and multi-sensor fusion. However, FPS was too low, and CPU usage was too high to use onboard. Results of the experiment can be seen in video clips here.4

C. STATE ESTIMATION

The EKF estimates a global map of the gates, and, since the gates are stationary, the gate detections align the VINS-Mono estimate with the global gate map, i.e., compensates for the VINS-Mono drift. The EKF has significantly improved the state estimation accuracy relative to the gates. The proposed EKF is not constrained only to use the next gate but can work with any gate detection and even profits from multiple detections in one image. The filtering result is shown in Fig. 10. The EKF runs the state prediction and measurement update loop and the estimated state. The error distributions between estimated states and planned path or ground-truth states are shown on the right side of Fig. 10. Fig. 13 also shows the estimated trajectory with the nominal performed trajectories of the quadrotor in a FlightGoggle drone racing arena.

D. PATH PLANNING

A typical drone race involves the challenge of flying through a series of checkpoints/gates and finishing the course within minimum time. We have tested the minimum snap trajectory

4https://youtu.be/XHDKgtOX7kc
generation process in 2D and 3D environments, as shown in Fig. 11.

The use of a minimal set of waypoints yields the paths typically composed of natural high-speed arcs in unconstrained regions of the environment while slowing in tight spaces to minimize snap around sharp corners. This process guarantees asymptotic convergence to a globally optimal solution provided by sampling-based approaches but returns superior paths in much shorter running times than a purely sampling-based approach. Minimum-snap polynomial splines have proven very effective as quadrotor trajectories since the vehicle’s motor commands and attitude accelerations are proportional to the path’s snap or fourth derivative. We show that this minimum-snap technique can be coupled with a cascaded controller that generates fast, graceful flight paths in cluttered environments while accounting for collisions of the resulting polynomial trajectories. In Fig. 12, we also show the velocity $\dot{v} = v$, acceleration $\ddot{x} = a$, jerk $\dot{x} = j$ and snap of quadrotor to make sure that they are continuous in time.

### E. CONTROL SYSTEM

To successfully navigate a race track, a drone has to continually sense and interpret its environment and be robust to cluttered and possibly dynamic track layouts. It needs precise control and estimation to support the aggressive maneuvers required to traverse a track at high speed. We attempted to solve these problems using a cascaded PID control for position and attitude. A cascade PID controller outputs the appropriate thrust and angular rates to the built-in low-level angular speeds controller. Fig. 13 shows the results of the proposed control algorithms. From the figure, we can observe that the result shows that the drone can complete the track of 11 gates with a speed of 7 m/s.

The methods and the results compared with our proposed method selected for this paper are given in Table 1. We compared our results in terms of deep learning (detection), Estimation, path planning, and control, as well as we compared the speed and race track time with other methods. Table 1 and Fig. 13 show that the proposed method has successfully guided the drone through tight racecourses, reaching a speed...
up to 7 m/s, and completed the race track in 18.5 seconds. Moreover, we showed in Table 1 that YOLOX and DeepSORT, and Model Predictive Controller (MPC) would be used in future work.

**TABLE 1. Methods and results comparison.**

| Deep Learning (Detection) | Estimation | Path Planning | Control | Speed (avg.) | Race Track Time |
|---------------------------|------------|---------------|---------|--------------|-----------------|
| DeepLabV3+, Xception, ResNet (Selected for Paper) | VIO | Polynomial | Cascaded P/T | 7m/s | 18.5 s |
| YOLO3 + ENet Network | Kalman Filter | Others | Linear PID | 6.6 s | not completed |
| YOLOX + DeepSORT (Future work) | Kalman Filter | Others | PID | 6.1 m/s | not completed |
| YOLOX + DeepSORT (Future work) | VIO | Polynomial | MPC | – | – |

While our current set of experiments was conducted in drone racing, we believe that the presented approach could have broader implications for building robust robot navigation systems that need to be able to act in a highly dynamic world.

**REFERENCES**

[1] M. Kazim, A. T. Azar, A. Koubaa, and A. Zaidi, “Disturbance-rejection-based optimized robust adaptive controllers for UAVs,” *IEEE Syst. J.*, vol. 15, no. 2, pp. 3097–3108, Jun. 2021.
[2] DRL. (2020). *AIRR* | The Drone Racing League. Accessed: Jan. 12, 2021. [Online]. Available: https://thedroneracingleague.com/airr/
[3] The New York Times. (2020). *Alphapilot AI Drone Racing League*. Accessed: Jan. 12, 2021. [Online]. Available: https://www.nytimes.com/2020/03/26/technology/alphapilot-ai-drone-racing.html
[4] H. Moon, Y. Sun, J. Baltes, and S. J. Kim, “The IROS 2016 competitions [competitions],” *IEEE Robot. Autom. Mag.*, vol. 24, no. 1, pp. 20–29, Mar. 2017.
[5] H. Moon, J. Martinez-Carranza, T. Cieslewski, M. Faessler, D. Falanga, A. Simovic, D. Scaramuzza, S. Li, M. Ozo, C. De Wager, G. de Croon, S. Hwang, S. Jung, H. Shim, H. Kim, M. Park, T.-C. Au, and S. J. Kim, “Challenges and implemented technologies used in autonomous drone racing,” *Intell. Service Robot.*, vol. 12, no. 2, pp. 137–148, Apr. 2019.
[6] C. Fu, R. Duan, D. Kircali, and E. Kayacan, “Onboard robust visual tracking for UAVs using a reliable global-local object model,” *Sensors.*, vol. 16, no. 9, p. 1406, Jul. 2016.
[7] M. Kazim, A. T. Azar, A. Koubaa, Z. F. Ibrahim, A. Zaidi, and L. Zhang, “Event-driven programming-based path planning and navigation of UAVs around a complex urban environment,” in *Unmanned Aerial Systems (Advances in Nonlinear Dynamics and Chaos)*, A. Koubaa and A. T. Azar, Eds. New York, NY, USA: Academic, 2021, ch. 21, pp. 531–565. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128202760000285
[8] S. Jung, S. Hwang, S. Hsin, and D. H. Shim, “Perception, guidance, and navigation for indoor autonomous drone racing using deep learning,” *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2539–2544, Jul. 2018.
[9] S.-Y. Shin, Y.-W. Kang, and Y.-G. Kim, “Obstacle avoidance drone by deep reinforcement learning and its racing with human pilot,” *Appl. Sci.*, vol. 9, no. 24, p. 5571, Dec. 2019.
[10] T. Morales, A. Sarabakh, and E. Kayacan, “Image generation for efficient neural network training in autonomous drone racing,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.
[11] D. Kim, H. Ryu, J. Yunchohor, and D. H. Shim, “A deep learning-aided automatic vision-based control approach for autonomous drone racing in game of drones competition,” in *Proc. NeurIPS Competition Demonstration Track*, 2020, pp. 37–46.
[12] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, “Learning monocular reactive UAV control in cluttered natural environments,” in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 1765–1772.
[13] M. Müller, G. Li, V. Casser, N. Smith, D. L. Michels, and B. Gianem, “Learning a controller fusion network by online trajectory filtering for vision-based UAV racing,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 1–9.
[14] J. Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” 2018, arXiv:1804.02767.
[15] The Drone Racing League. (2020). *AlphaPilot—Lockheed Martin AI Drone Racing League*. Accessed: Jan. 20, 2021. [Online]. Available: https://www.herox.com/alphapilot/85-2019-virtual-qualifier-tests
[16] Intel. (2021). *Intel Aero Ready to Fly Drone*. Accessed: Jan. 2, 2021. [Online]. Available: https://www.intel.com/content/www/us/en/support/products/97174/drones/development-drones/intel-aero-products.html
[17] Z. Zuo, “Trajectory tracking control design with command-filtered compensation for a quadrotor,” *IET Control Theory Appl.*, vol. 4, no. 11, pp. 2343–2355, Nov. 2010.
[18] Z. Zuo, “Adaptive trajectory tracking control design with command filtered compensation for a quadrotor,” *J. Vib. Control*, vol. 19, no. 1, pp. 94–108, Jan. 2013.
[19] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, “Deep drone racing: Learning agile flight in dynamic environments,” 2018, arXiv:1806.08548.
MUHAMMAD KAZIM (Member, IEEE) received the B.S. degree in electronics and the M.S. degree in electrical engineering from COMSATS University Islamabad, Islamabad, Pakistan, in 2013 and 2015, respectively, and the Ph.D. degree in control science and engineering from the Harbin Institute of Technology, Harbin, China, in December 2021. He was a Visiting Research Scholar with Prince University, Riyadh, Saudi Arabia, from September 2019 to February 2020 and January 2021 to July 2021. His research interests include control theory, autonomous aerial and ground vehicles, robotics, and RL.

ADEEL ZAIDI (Member, IEEE) received the B.S. degree in electronics engineering from the COMSATS University of Information Technology, Islamabad, Pakistan, in 2012, and the M.S. degree in electrical (control) engineering from the National University of Sciences and Technology, Islamabad, in 2015. He is currently pursuing the Ph.D. degree in control science and engineering with the Harbin Institute of Technology, Harbin, China. His research interests include multi-agent control, simulation, and control of dynamic systems; mobile robots; autonomous vehicles; computer vision; and machine learning.

SHAFQAT ALI received the B.E. degree in electronic engineering from the Dauowd University of Engineering and Technology, Karachi, Pakistan, and the M.Sc. degree in electrical engineering specialized in control systems from the University of Engineering and Technology, Lahore, Pakistan, in 2013 and 2017, respectively. Currently, he is pursuing the Ph.D. degree in control science and engineering with the Harbin Institute of Technology, Harbin, China. His research interests include robust control, fault diagnosis, switched systems, and sliding mode observer.

MUHAMMAD TASKEEN RAZA received the B.Sc. and M.Sc. degrees in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2004 and 2011, respectively, and the Ph.D. degree from the Department of Electrical Engineering, Pakistan Institute of Engineering and Applied Sciences, Islamabad, Pakistan, in 2016. He has been with the Department of Electrical Engineering, Lahore College for Women University (LCWU), Lahore, since 2006, where he is an Assistant Professor. His current research interests include fault diagnosis; fault tolerant control; autonomous systems; disease diagnosis and prognosis using machine learning, deep learning, data science, and data analytics; the Internet of Things (IoT); and the Internet of Medical Things (IoMT).

GHULAM ABBAS (Senior Member, IEEE) received the B.E. degree from the University of Engineering and Technology, Lahore, Pakistan, in 2004, and the M.E. and Ph.D. degrees from the Institut National des Sciences Appliquées de Lyon (INSA Lyon), France, in 2008 and 2012, respectively, all in electrical engineering. He is currently working as an Associate Professor with the Department of Electrical Engineering, The University of Lahore, Lahore. He has published a number of papers in various international and national IEEE conferences and journals of repute like IEEE, Elsevier, and MDPI. His research interests include analog and digital controller designs for power switching converters, soft computing, and power systems optimization. He is a Reviewer of a number of premier conferences and journals, including the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, Asian Journal of Control, Engineering Optimization, and IEEE ACCESS. He has served several electrical and computer engineering journals as a guest editor.

NASIM ULLAH (Member, IEEE) received the Ph.D. degree in mechatronic engineering from Beihang University, Beijing, China, in 2013. From September 2006 to September 2010, he was a Senior Design Engineer with industry in Pakistan. He is currently working as a Full Professor of electrical engineering with Taif University, Saudi Arabia. His research interests include renewable energy, flight control systems, integer and fractional order modeling of dynamic systems, integer/fractional-order adaptive robust control methods, fuzzy/NN, hydraulic and electrical servos, and epidemic and vaccination control strategies.

AHMAD AZIZ AL-AHMADI (Member, IEEE) received the Ph.D. degree in nano-electronic devices from the School of Engineering, Ohio University, Athens, OH, USA. Since July 2017, he has been the Dean of the Faculty of Engineering, Taif University, Saudi Arabia. He is currently an Associate Professor with the Department of Electrical Engineering, Faculty of Engineering, Taif University. His research interests include control systems and nano-electronic devices.