**Linear Quadratic Integral Differential Game applied to the Real-time Control of a Quadrotor Experimental setup**

Hadi Nobahari\*, Alireza Sharifi

Sharif University of Technology, Zip Code 1458889694, Tehran, Iran

nobahari@sharif.edu, alireza\_sharifi@ae.sharif.edu

**Abstract.**

The accurate attitude control of a quadrotor is necessary, especially when facing disturbance. In this study, a linear quadratic with integral action based on the differential game theory is implemented on a quadrotor experimental setup. A continuous state-space model of the setup is derived using the linearization of nonlinear equations of motion, and its parameters are identified with the experimental results. Then, the attitude control commands of the quadrotor are derived based on two players; one finds the best attitude control command, and the other creates the disturbance by mini-maximizing a quadratic criterion, defined as the sum of outputs plus the weighted control effort and disturbance. The performance of the proposed structure is investigated in level flight and compared to the linear quadratic regulator controller. Results demonstrate that the proposed approach has an excellent performance in dissipating the disturbances.

**Keywords:** Linear Quadratic Differential Game, Quadrotor, Real-time, 3DoF Experimental setup, Optimal Control, Robust Control.

\*Corresponding Author.

1. Introduction

A quadrotor is a type of helicopter with four rotors that plays a significant role in today's society, including research, military, imaging, recreation, and agriculture. The performance of the quadrotor relies on the control system, including attitude, altitude, and position subsystems. In the attitude control of the quadrotor, it is vital to maintain the attitude outputs at the desired level using control commands such as the rotational speed of the rotors, when the disturbances occur suddenly. Therefore, much research is being conducted on the automatic control of the attitudes' quadrotor in facing the disturbance.

In \cite{PID}, a Proportional Integral Derivative (PID) controller is used to regulate the quadrotor attitude. However, the control objectives have not been effectively achieved with this controllers when the disturbance occurs. To solve this problem the model-based approaches \cite{model\_base} are utilized for controller design. These controllers work based on information from the quadrotor's attitude model and disturbance to produce the best control command.

Various model-based controllers can be found within the literature, the most well-known of which are intelligent control, the nonlinear control, robust control, and optimal control to reduce the disturbance effect in the attitude control and provide a faster control algorithm in facing the modeling error. In the intelligent controller category, the artificial intelligence computing approaches like fuzzy logic \cite{fuzzy}, iterative learning \cite{iterative\_Learning}, machine learning \cite{machine\_learning}, reinforcement learning \cite{Reinforcement\_Learning}, and evolutionary computation \cite{Evolutionary} have been utilized to regulate the quadrotor's attitude.

Moreover, nonlinear control methods such as Feedback Linearization (FBL) \cite{FBL} and Sliding Mode Control (SMC) \cite{SMC} have been applied to control the roll, pitch, and yaw angles of the quadrotor. In the optimal controller category, a Linear Quadratic Regulator (LQR) \cite{LQR} and Linear Quadratic Gaussian (LQG) \cite{LQG} have been implemented on the quadrotor based on the minimization of a quadratic criterion, including regulation performance and control effort to provide optimally controlled feedback gains.

Linear Quadratic Regulator Differential Game (LQR-DG) control approach \cite{LQDG, robust\_LQDG} is a class of optimal and robust controller methods that controls the outputs of a system based on its linear model and mini-maximization of a cost function. This approach has been utilized to stabilize and control various nonlinear and complex systems such as a ship controller \cite{LQDG\_ship}. Moreover, in the LQR-DG control method, the control commands are analytically generated based on a pursuit-evasion of two players, one tracks the best control command, and the other creates the disturbance. This is one of the distinctive features of the LQR-DG controller and an important difference from other optimal control methods.

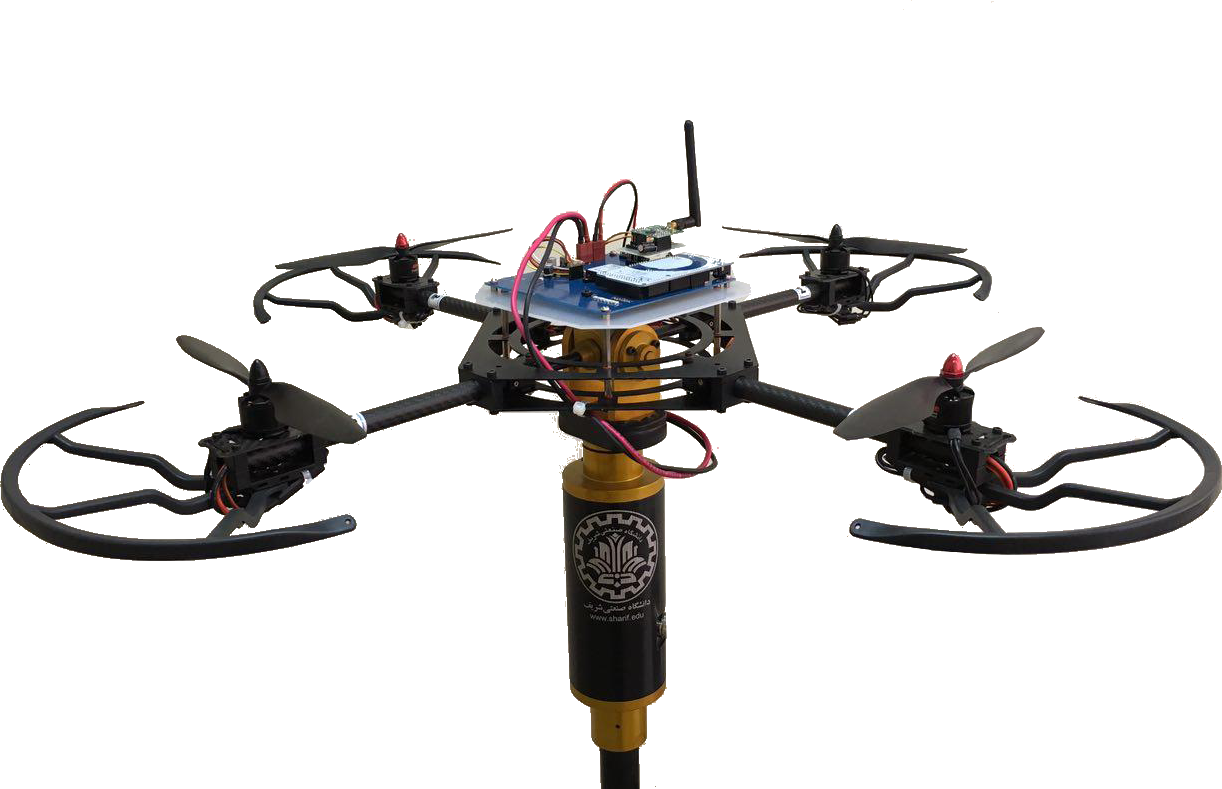
In this study, an LQR controller method based on the differential game theory, with an integral action called Linear Quadratic Integral Regulator Differential Game (LQIR-DG) controller, is proposed to generate the most efficient control command for an experimental setup of the quadrotor when facing the disturbance. Since the LQIR-DG is affected by an accurate model of the system, first, the dynamic of the three-degree-of-freedom setup of the quadrotor is modeled. Then, the linear state-space form the quadrotor model is extracted using the linearization of the nonlinear equations of motion to utilize in the proposed control problem. Moreover, the model's parameters are identified and verified against the experimental values. Next, the LQIR-DG technique is applied to the experimental setup of the quadrotor to reduce the effect of disturbance. The performance of the suggested controller is examined when the disturbance occurs. The results show the successful performance of the LQIR-DG scheme in reducing the disturbance.

In the remainder of this study, the problem is defined in section \ref{sec:problem\_statement}. The dynamics model for the experimental setup of the quadrotor is derived in details, in section \ref{sec:modeling}. In section \ref{sec: diff game}, The LQIR-DG architecture is denoted. Finally, in sections \ref{sec:results} and \ref{sec:conclusion}, Numerical results and conclusion are provided, respectively.

1. Problem Statement

Here, the model of the three-degree-of-freedom setup of the quadrotor is presented in details.

Here, a nonlinear dynamic is presented for the setup of the quadrotor, as illustrated in \figurename{\ref{quadlab}}. The quadrotor is free to rotate about its roll, pitch, and yaw axes. The Euler angles and angular velocities along three orthogonal axes are measured simultaneously using the Attitude and Heading Reference Systems (AHRS). These noisy measurements are utilizes in the LQIR-DG for control of the Euler angles. The block diagram of the controller structure is illustrated in \figurename{\ref{block\_diagram}}.



**Fig. 7.** 3DoF setup of the quadrotor

1. Modeling of the Quadrotor Setup

Here, the model of the three-degree-of-freedom setup of the quadrotor is presented in details. For this purpose, first, the configuration of the quadrotor is denoted. Then, the nonlinear model of the attitude dynamics is derived to denote the state-space form. Finally, the nonlinear model is linearized to utilize in the control purposes.

* 1. Configuration of the Quadrotor

\figurename{\ref{QuadAssum}} denotes the quadrotor schematic. Each rotor has an angular velocity ,$\Omega\_i$, rotating about the $z\_B$ axis in the body coordinate system. Rotors 1 and 3 rotate counterclockwise, while rotors 2 and 4 rotate clockwise, to cancel yawing moment.

* 1. Dynamic Model

The quadrotor kinetic model, derived using the Newton-Euler method, is stated as \cite{b15}, \cite{b16}

\begin{align}

&\dot p = \dfrac{\mathrm{I}\_{\text{yy}} - \mathrm{I}\_{\text{zz}}}{\mathrm{I}\_{\text{xx}}} qr + q \dfrac{\mathrm{I}\_{\text{rotor}}}{\mathrm{I}\_{\text{xx}}}\Omega\_r + \dfrac{u\_{\text{roll}}}{\mathrm{I}\_{\text{xx}}} + \dfrac{d\_{\text{roll}}}{\mathrm{I}\_{\text{xx}}}

\\

&\dot q = \dfrac{\mathrm{I}\_{\text{zz}} - \mathrm{I}\_{\text{zz}}}{\mathrm{I}\_{\text{yy}}} rp + p \dfrac{\mathrm{I}\_{\text{rotor}}}{\mathrm{I}\_{\text{xx}}}\Omega\_r + \dfrac{u\_{\text{pitch}}}{\mathrm{I}\_{\text{yy}}} + \dfrac{d\_{\text{pitch}}}{\mathrm{I}\_{\text{yy}}}

\\

&\dot r = \dfrac{\mathrm{I}\_{\text{xx}} - \mathrm{I}\_{\text{yy}}}{\mathrm{I}\_{\text{zz}}} pq + \dfrac{u\_{\text{yaw}}}{\mathrm{I}\_{\text{zz}}} + \dfrac{d\_{\text{yaw}}}{\mathrm{I}\_{\text{zz}}}

\end{align}

where $(p, q, r)$ are the angular velocities. $d\_{\text{roll}}$, $d\_{\text{pitch}}$, and $d\_{\text{yaw}}$ are the disturbances, generated in $x\_B$, $y\_B$ and $z\_B$, respectively. Moreover, $\mathrm{I}\_{\text{xx}}$, $\mathrm{I}\_{\text{yy}}$, and $\mathrm{I}\_{\text{zz}}$ are the principal moment of inertia and $\mathrm{I}\_{\text{rotor}}$ is a rotor inertia about its axis. The relation between the angular body rates and the Euler angles rates are obtained as

\begin{align}

\dot\phi &= p + (q\sin(\phi) + r\cos(\phi))\tan(\theta)\\

\dot \theta &= q\cos(\phi) - r\sin(\phi)\\

\dot\psi &= (q\sin(\phi) + r\cos(\phi))/{\cos(\theta)}\\

\end{align}

where $(\phi, \theta, \psi)$ are roll, pitch, and yaw angles.

Moreover, $\Omega\_r$, called the overall residual rotor angular velocity, is computed as

\begin{equation}

\Omega\_r = -\Omega\_1 + \Omega\_2 - \Omega\_3 + \Omega\_4

\end{equation}

* 1. Control Commands

The control inputs $u\_{\text{roll}}$, $u\_{\text{pitch}}$, and $u\_{\text{yaw}}$ are roll, pitch, and yaw moments, obtained from the rotors, defined as

\begin{align}

&u\_{\text{roll}} = \mathrm{b\,d}\_{\text{cg}} (\Omega\_2^2 - \Omega\_4^2)\\

&u\_{\text{pitch}} = \mathrm{b\,d}\_{\text{cg}} (\Omega\_1^2 - \Omega\_3^2) \\

&u\_{\text{yaw}} = \mathrm{d} (\Omega\_1^2 - \Omega\_2^2 + \Omega\_3^2 - \Omega\_4^2)\\

\end{align}

Also, d and b are, respectively, drag and thrust coefficients. $\mathrm{d}\_{\text{cg}}$ is the distance of rotors from the gravity center . Hence, the angular velocity commands are obtained as

\begin{align}

\Omega\_{c, 1}^2 &= \Omega\_{\text{mean}}^2 + \dfrac{1}{2\mathrm{b\,d}\_{\text{cg}}}u\_{\text{pitch}} + \dfrac{1}{4d}u\_{\text{yaw}} \\

\Omega\_{c, 2}^2 &= \Omega\_{\text{mean}}^2 + \dfrac{1}{2\mathrm{b\,d}\_{\text{cg}}}u\_{\text{roll}} - \dfrac{1}{4d}u\_{\text{yaw}}\\

\Omega\_{c, 3}^2 &= \Omega\_{\text{mean}}^2 - \dfrac{1}{2\mathrm{b\,d}\_{\text{cg}}}u\_{\text{pitch}} + \dfrac{1}{4d}u\_{\text{yaw}} \\

\Omega\_{c, 4}^2 &= \Omega\_{\text{mean}}^2 - \dfrac{1}{2\mathrm{b\,d}\_{\text{cg}}}u\_{\text{roll}} - \dfrac{1}{4d}u\_{\text{yaw}}

\end{align}

where $\Omega\_{\text{mean}}$ is the nominal of the rotor angular velocities.

* 1. State-Space Form

Here, the state-space model is presented for the control purposes. By defining $x\_1 = p$, $x\_2 = q$, $x\_3 = r$, $x\_4 = \phi$, $x\_5 = \theta$, and $x\_6 = \psi$; the model of in state-space form are denoted as

\begin{align}\label{eq:diffeq}

\dot x\_1 &= \dfrac{\mathrm{I}\_{\text{yy}} - \mathrm{I}\_{\text{zz}}}{\mathrm{I}\_{\text{xx}}} x\_2 x\_3 + x\_2 \dfrac{\mathrm{I}\_{\text{rotor}}}{\mathrm{I}\_{\text{xx}}}\Omega\_r + \dfrac{u\_{\text{roll}}}{\mathrm{I}\_{\text{xx}}} + \dfrac{d\_{\text{roll}}}{\mathrm{I}\_{\text{xx}}} \\

\dot x\_2 &= \dfrac{\mathrm{I}\_{\text{zz}} - \mathrm{I}\_{\text{zz}}}{\mathrm{I}\_{\text{yy}}} x\_1 x\_3 - x\_1 \dfrac{\mathrm{I}\_{\text{rotor}}}{\mathrm{I}\_{\text{xx}}}\Omega\_r + \dfrac{u\_{\text{pitch}}}{\mathrm{I}\_{\text{yy}}} + \dfrac{d\_{\text{pitch}}}{\mathrm{I}\_{\text{yy}}}\\

\dot x\_3 &= \dfrac{\mathrm{I}\_{\text{xx}} - \mathrm{I}\_{\text{yy}}}{\mathrm{I}\_{\text{zz}}} x\_1 x\_2 + \dfrac{u\_{\text{yaw}}}{\mathrm{I}\_{\text{zz}}} + \dfrac{d\_{\text{yaw}}}{\mathrm{I}\_{\text{zz}}}\\

\dot x\_4 &= x\_1 + (x\_2\sin(x\_4) + x\_3\cos(x\_4))\tan(x\_5)

\\

\dot x\_5 &= x\_2\cos(x\_4) - x\_3\sin(x\_4)\\

\dot x\_6 &= (x\_2\sin(x\_4) + x\_3\cos(x\_4))/\cos(x\_5) \label{eq:diffeq-end}

\end{align}

The measurement model is written as

\begin{equation}

\begin{split}

\boldsymbol{\mathrm{z}} &= \begin{bmatrix}

p\_m & q\_m & r\_m & \phi\_m & \theta\_m & \psi\_m

\end{bmatrix}^\mathrm{T}

\end{split}

\end{equation}

* 1. Linear Model

The continuous-time linear model is utilized to drive the control commands on the quadrotor. The linear state-space model is denoted as

\begin{equation}\label{eq:linear}

\boldsymbol{\dot{\mathrm{x}}}(t) = \boldsymbol{\mathrm{Ax}}(t) + \boldsymbol{\mathrm{Bu}}(t) + \boldsymbol{\mathrm{B\_{d}d}}(t)

\end{equation}

where $\boldsymbol{\mathrm{A}}$, $\boldsymbol{\mathrm{B}}$, and $\boldsymbol{\mathrm{B\_d}}$ are the system, input and disturbance matrices, respectively. Moreover, $\boldsymbol{\mathrm{d}}$ is the disturbance. The measurements equation is stated as

\begin{equation}

\boldsymbol{{\mathrm{z}}}(t) = \boldsymbol{\mathrm{x}}(t)

\end{equation}

According to Eqs.\eqref{eq:diffeq}-\eqref{eq:diffeq-end}, the linear dynamic model around the equilibrium points $(\boldsymbol{{\mathrm{x}}}\_e\!=\!0 \text{ and } \boldsymbol{{\mathrm{u}}}\_e\!=\!0)$ of the quadrotor setup is denoted as

\begin{equation}

\begin{split}

\boldsymbol{{\mathrm{\dot x}}} = \begin{bmatrix}

\boldsymbol{{\mathrm{\dot x\_{\text{roll}}}}}\\

\boldsymbol{{\mathrm{\dot x\_{\text{pitch}}}}}\\

\boldsymbol{{\mathrm{\dot x\_{\text{yaw}}}}}

\end{bmatrix} &= \begin{bmatrix}

\boldsymbol{{\mathrm{A\_{\text{roll}}}}} & \boldsymbol{0} & \boldsymbol{0}\\

\boldsymbol{0} & \boldsymbol{{\mathrm{A\_{\text{pitch}}}}} & \boldsymbol{0} \\

\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{{\mathrm{A\_{\text{yaw}}}}}

\end{bmatrix} \begin{bmatrix}

\boldsymbol{{\mathrm{x\_{\text{roll}}}}}\\

\boldsymbol{{\mathrm{x\_{\text{pitch}}}}}\\

\boldsymbol{{\mathrm{x\_{\text{yaw}}}}}

\end{bmatrix}

\\[1em]

& + \begin{bmatrix}

\boldsymbol{{\mathrm{B\_{\text{roll}}}}} & \boldsymbol{0} & \boldsymbol{0}\\

\boldsymbol{0} & \boldsymbol{{\mathrm{B\_{\text{pitch}}}}} & \boldsymbol{0} \\

\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{{\mathrm{B\_{\text{yaw}}}}}

\end{bmatrix}

\begin{bmatrix}

\boldsymbol{{\mathrm{u\_{\text{roll}}}}}\\

\boldsymbol{{\mathrm{u\_{\text{pitch}}}}}\\

\boldsymbol{{\mathrm{u\_{\text{yaw}}}}}

\end{bmatrix}\\[1em]

& + \begin{bmatrix}

\boldsymbol{{\mathrm{B\_{\text{roll}}}}} & \boldsymbol{0} & \boldsymbol{0}\\

\boldsymbol{0} & \boldsymbol{{\mathrm{B\_{\text{pitch}}}}} & \boldsymbol{0} \\

\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{{\mathrm{B\_{\text{yaw}}}}}

\end{bmatrix} \begin{bmatrix}

\boldsymbol{{\mathrm{d\_{\text{roll}}}}}\\

\boldsymbol{{\mathrm{d\_{\text{pitch}}}}}\\

\boldsymbol{{\mathrm{d\_{\text{yaw}}}}}

\end{bmatrix}

\end{split}

\end{equation}

where $\boldsymbol{\mathrm{x}}\_{\text{roll}} = \begin{bmatrix}

p & \phi

\end{bmatrix}^\mathrm{T}$, $\boldsymbol{\mathrm{x}}\_{\text{pitch}} = \begin{bmatrix}

q & \theta \end{bmatrix}^\mathrm{T}$, and $\boldsymbol{\mathrm{x}}\_{\text{yaw}} = \begin{bmatrix}

r & \psi

\end{bmatrix}^\mathrm{T}$.

Moreover, the state and input matrices are presented as

\begin{equation}

\begin{split}

\boldsymbol{\mathrm{A}}\_{\text{roll}} =\boldsymbol{\mathrm{A}}\_{\text{pitch}} = \boldsymbol{\mathrm{A}}\_{\text{yaw}} = \begin{bmatrix}

0 & 0\\

1 & 0

\end{bmatrix}

\end{split}

\end{equation}

\begin{equation}

\begin{split}

\boldsymbol{\mathrm{B}}\_{\text{roll}} = \begin{bmatrix}

\dfrac{1}{\mathrm{I}\_{\text{xx}}}

\\[1em]

0

\end{bmatrix};~ \boldsymbol{\mathrm{B}}\_{\text{pitch}} = \begin{bmatrix}

\dfrac{1}{\mathrm{I}\_{\text{yy}}}

\\[1em]

0

\end{bmatrix};~ \boldsymbol{\mathrm{B}}\_{\text{yaw}} = \begin{bmatrix}

\dfrac{1}{\mathrm{I}\_{\text{zz}}}

\\[1em]

0

\end{bmatrix}

\end{split}

\end{equation}

1. Formulation of the Controller Design

In the LQIR-DG controller structure, an integral action is added to the LQR-DG controller to cancel the steady-state errors for reference tracking. For this purpose, first, the augmented state space of the linear quadrotor model is defined to utilize in the controller architecture. Then, the LQR-DG controller design procedure is presented to produce the best control commands for the experimental setup of the quadrotor.

* 1. Augmented State Space Formulation

To add the integral action to the controller structure, the augmented states are defined as follows:

\begin{equation}\label{lqidg\_x}

\boldsymbol{\mathrm{x\_{a\_i}}} = \begin{bmatrix}

\boldsymbol{\mathrm{x\_i}} &

\displaystyle \int \boldsymbol{\mathrm{x\_i}}

\end{bmatrix}^\mathrm{T}

\end{equation}

where $i$ = roll, pitch, and yaw.

Then, the quadrotor dynamics model, denoted by Eq.\eqref{eq:linear}, is denoted in the augmented state-space model as

\begin{equation}\label{systemlqidg}

\begin{split}

\boldsymbol{\dot{\mathrm{x}}\_a}(t) &= \boldsymbol{\mathrm{A\_ax\_a}}(t) + \boldsymbol{\mathrm{B\_{{a}}u}}(t) + \boldsymbol{\mathrm{B\_{{d\_a}}d}}(t)%, \quad \boldsymbol{x}(0) =

\end{split}

\end{equation}

where matrices $\boldsymbol{\mathrm{A\_a}}$ and $\boldsymbol{\mathrm{B\_a}}$ are defined as follows:

\begin{equation}

\boldsymbol{\mathrm{A\_a}} = \begin{bmatrix}

\boldsymbol{\mathrm{A}} & \boldsymbol{0}\\

\boldsymbol{\mathrm{I}} & \boldsymbol{0}

\end{bmatrix}

\end{equation}

\begin{equation}

\boldsymbol{\mathrm{B\_a}} = \boldsymbol{\mathrm{B\_{{d\_a}}}} = \begin{bmatrix}

\boldsymbol{\mathrm{B}}\\

\boldsymbol{0}

\end{bmatrix}

\end{equation}

In the above equation $\boldsymbol{\mathrm{I}}$ denotes the identity matrix.

\subsection{LQIR-DG Controller Method}

\noindent The LQIR-DG controller is an optimal and robust method based on the differential game theory. This controller consists of two essential players: one finds the best control command, and the other creates the worst disturbance.

For this purpose, the first player tries to minimize a cost function; while the second player is assumed to maximize it. Therefore, the quadratic cost function equation is denoted using min-max operators as follows:

\begin{equation}

\begin{split}

&\min\_{u} \max\_{d} J(\boldsymbol{\mathrm{x\_{a\_i}}}, {u\_i}, {d\_i}) = J(\boldsymbol{\mathrm{x\_{a\_i}}}, {u^\*\_i}, {d^\*\_i})= \\ & ~~\min\_{u} \max\_{d}

\int\_{0}^{\mathrm{t\_f}}\biggl (\boldsymbol{\mathrm{x^\mathrm{T}\_{a\_i}}} \boldsymbol{\mathrm{Q\_i}} \boldsymbol{\mathrm{x\_{a\_i}}}+

{{u^\mathrm{T}\_i}} {{R}} {{u\_i}}-

{{d^\mathrm{T}\_{i}}} {{ R\_{d} d\_{i}}}

\biggl )\mathrm{d}t

\end{split}

\end{equation}

where ${{ R}}$ and ${{R\_{d}}}$ are symmetric nonnegative definite matrices and $\boldsymbol{\mathrm{Q\_i}} $ is a symmetric positive definite matrix. Moreover, $\mathrm{t\_f}$ is the final time. To solve this problem, connections between the general optimal problem and the LQIR problem are considered \cite{LQDG} and consequently the optimum control effort is computed for the each control loop as follows:

\begin{equation}

\begin{split}

{{u\_i}}(t) = -\boldsymbol{{\mathrm{K}}\_{i}}(t) \boldsymbol{{\mathrm{x\_{a\_i}}}}(t)

\end{split}

\end{equation}

\begin{equation}

{{d\_i}}(t) = \boldsymbol{{\mathrm{K}}\_{i}}(t)\boldsymbol{{\mathrm{x\_{a\_i}}}}(t)

\end{equation}

where $\boldsymbol{{\mathrm{K\_i}}}$ and $\boldsymbol{{\mathrm{K\_{d\_i}}}}$ are a time varying gain, given by

\begin{equation}

\boldsymbol{{\mathrm{K\_i}}} = {{{R}}^{-1}}\boldsymbol{{\mathrm{B}\_{a\_i}^\mathrm{T}}}\boldsymbol{{\mathrm{P}}\_{a\_i}}(t)

\end{equation}

\begin{equation}

\boldsymbol{{\mathrm{K\_{d\_i}}}} = {{{R}}^{-1}\_{d}}\boldsymbol{{\mathrm{B}\_{a\_{d\_i}}^\mathrm{T}}}\boldsymbol{{\mathrm{P}}\_{a\_{d\_i}}}(t)

\end{equation}

where $\boldsymbol{{\mathrm{P}}\_{a\_i}}(t)$ and $\boldsymbol{{\mathrm{P}}\_{a\_{d\_i}}}(t)$ satisfy

\begin{equation}\label{coupled\_riccatti\_LQIDG}

\begin{split}

\boldsymbol{\dot{\mathrm{P}}\_{a\_i}}&(t) = -\boldsymbol{\mathrm{A^\mathrm{T}\_a}}\boldsymbol{\mathrm{P\_{a\_i}}}(t) - \boldsymbol{\mathrm{P\_{a\_i}}}(t)\boldsymbol{\mathrm{A\_a}} - \boldsymbol{\mathrm{Q\_i}} +\\ &\boldsymbol{\mathrm{P\_{a\_i}}}(t)\boldsymbol{\mathrm{S\_{a\_i}}}(t)\boldsymbol{\mathrm{P\_{a\_i}}}(t) + \boldsymbol{\mathrm{P\_{a\_i}}}(t)\boldsymbol{\mathrm{S\_{a\_{d\_i}}}}(t)\boldsymbol{\mathrm{P\_{a\_{d\_i}}}}(t)

\end{split}

\end{equation}

\begin{equation}

\begin{split}

\boldsymbol{\dot{\mathrm{P}}\_{a\_{d\_i}}}&(t) = -\boldsymbol{\mathrm{A^\mathrm{T}\_a}}\boldsymbol{\mathrm{P\_{a\_{d\_i}}}}(t) - \boldsymbol{\mathrm{P\_{a\_{d\_i}}}}(t)\boldsymbol{\mathrm{A\_a}} - \boldsymbol{\mathrm{Q\_{i}}} +\\ &\boldsymbol{\mathrm{P\_{a\_{d\_i}}}}(t)\boldsymbol{\mathrm{S\_{a\_{d\_i}}}}(t)\boldsymbol{\mathrm{P\_{a\_{d\_i}}}}(t) + \boldsymbol{\mathrm{P\_{a\_{d\_i}}}}(t)\boldsymbol{\mathrm{S\_{a\_i}}}(t)\boldsymbol{\mathrm{P\_{a\_i}}}(t)

\end{split}

\end{equation}

where $\boldsymbol{\mathrm{S\_{a\_i}}} = \boldsymbol{\mathrm{B\_{a\_i}}}R^{-1}\boldsymbol{\mathrm{B}^\mathrm{T}\_{a\_i}}$ and $\boldsymbol{\mathrm{S\_{a\_{d\_i}}}} = \boldsymbol{\mathrm{B}\_{a\_{d\_i}}}R\_{d}^{-1}\boldsymbol{\mathrm{B}^\mathrm{T}\_{a\_{d\_i}}}$.

In this study, the steady-state values of the above equations $(\boldsymbol{{\mathrm{P}}} \text{ as } \mathrm{t\_f} \to \infty)$ are utilized to generate a feedback control law.

1. Result and Discussion

Here, the results of the LQIR-DG controller method are devoted to the control loops of the roll, pitch, and yaw of the experimental setup of the quadrotor. First, the controller parameters are tuned using the results of numerical simulations. Moreover, the performance of the LQIR-DG controller is compared to an LQR control strategy. The quadrotor parameters are shown in table \ref{tab:parameters}.

At the beginning of the inner loop, the priori position of the ants and the associated covariance are propagated. Next, the posteriori position of the ants is computed using EKF to estimate the current output. The estimated outputs are compared with the real measurement and each ant is assigned a cost, based on the quality of its estimation. Ants use their experience to update the pheromone distribution over the continuous state space. As in CACF ‎[3], a normal function is utilized to model the pheromone distribution. Ants use this pheromone distribution to move from their current position toward the minimum cost estimation. The normal distribution permits all points of the state space to be chosen, either close to or far from the best point. The inner loop is terminated after a predefined number of iterations. Finally, the current state is estimated using a mean operator. In the following, these steps are discussed in detail.

* 1. Initialization

ECACF has some control parameters that must be set before the execution of the algorithm including the number of ants, N, and the number of top ants, Nt. Moreover, the initial position of ant *j*, , is randomly generated and the associated covariance, , is initialized for *j*=1,…,N.

* 1. Propagation of State and Covariance

At the beginning of the *i*th iteration of the inner loop, the priori position of ant *j* at time *k*-1, defined as , is propagated using the stochastic process model as follows:

|  |  |
| --- | --- |
|  | (3) |

where the process noise vector  is randomly generated based on the known PDF of . Moreover, the priori covariance of the estimation error for ant *j* at time *k*-1, defined as , is propagated based on EKF as follows:

|  |  |
| --- | --- |
|  | (4) |

where  is the state transition matrix, defined as

|  |  |
| --- | --- |
|  | (5) |

* 1. Measurement Update

First, the posteriori position of ant *j* at time *k*, , is computed as a linear combination of the priori position of ant *j*, , and the difference between the actual measurement at time *k*, , and the predicted measurement, , according to the measurement-update equation of EKF as follows:

|  |  |
| --- | --- |
|  | (6) |

where  is calculated as

|  |  |
| --- | --- |
|  | (7) |

and  is the Kalman gain for ant *j* at time *k*, obtained as

|  |  |
| --- | --- |
|  | (8) |

Also,  is the observation matrix, defined as

|  |  |
| --- | --- |
|  | (9) |

Then, the current output, estimated in iteration *i* by ant *j* at time *k*, is calculated as

|  |  |
| --- | --- |
|  | (10) |

* 1. Evaluation of Cost Function

Each ant is assigned a cost, based on the quality of its current estimation. The cost function is defined as the square error between the real measurement, , and the estimated output, . Therefore, the cost, assigned in iteration *i* to ant *j* at time *k*, is calculated as

|  |  |
| --- | --- |
|  | (11) |

In this way‎, different points of the state space are evaluated and some knowledge about the state estimation problem is acquired.

* 1. Movement of the Ants

During any iteration, ants move from their current position to their destination using the current pheromone distribution. As in CACF ‎[3], pheromone distribution is modeled by a normal distribution function, given by

|  |  |
| --- | --- |
|  | (12) |

where  is the best point, found up to the *i*th iteration, at time *k* and n is the number of the state variables. Moreover,  is calculated as follows:

|  |  |
| --- | --- |
|  | (13) |

where  is the variance of the normal pheromone distribution function, defined for dimension *p*. This parameter is updated using the concept of weighted variance, proposed in ‎[22]:

|  |  |
| --- | --- |
|  | (14) |

where  and  are respectively the component of  and  in dimension *p*. This strategy means that the center of the normal pheromone distribution is the position of the best ant and the narrowness of its width depends on the aggregation of other competitors around the best one [22].

* 1. Stopping Condition

ECACF has two loops, each with its own specific stopping condition. The inner loop stops when the maximum number of iterations, *i*max, is reached. The outer loop stops when the measurements are finished.

* 1. States Estimation

After the termination of the inner loop, the states are estimated based on the average position of Nt top ants as follows:

|  |  |
| --- | --- |
|  | (15) |

where Nt denotes the number of top ants. In addition, the covariance matrix is calculated using Monte Carlo integration as

|  |  |
| --- | --- |
|  | (16) |

|  |
| --- |
| Set the number of ants, N, and the number of top ants, Nt.  Initialize randomly the position of the ants, , for *j*[1,N].  Initialize the covariance of the ants, , for *j*[1,N].  **While** (Measurements are available)  **For** *i*=1 to *i*max  **For** *j*=1 to N  Propagate the priori position of ant *j*, , according to “(3)”.  Compute the priori covariance, , according to “(4)”.  Compute the posteriori position of ant *j*, , according to “(6)”.  Estimate the current output, , according to “(10)”.  Compute the cost function, , according to “(11)”.  **Next** ant *j*  **For** *p*=1 to n  Compute the variance of the pheromone distribution, , according to “(14)”.  **Next** dimension *p*  Update the covariance matrix of the pheromone distribution, , according to “(13)”.  **For** *j*=1 to N  Move ants to the new destination, , using the normal PDF according to “(12)”.  .  .  **Next** ant *j*  **Next** iteration *i*  Sort ants according to the cost.  Compute the mean position of Nt top ants, , according to “(15)”.  Compute covariance, , according to “(16)”.  **For** *j*=1 to N  .  .  **Next** ant *j*  **Next** time step *k* |

**Fig. 2.** ECACF Pseudo-Code

1. Results and Discussion

In this section, the performance of the ECACF is investigated in several problems. First, it is compared with the EPF, UPF and CACF for a given benchmark. Then, the ECACF is utilized for simultaneous estimation of the states and the wind disturbances in a UAV. In this case, the ECACF is also verified through a PIL experiment. Finally, the real-time attitude estimation of a 3DOF experimental quadrotor is performed using the ECACF.

* 1. Benchmark Problem

In this section, a nonlinear single variable economic model [10, 11] is employed to examine the performance of the ECACF and compare it with other filters including EPF, UPF and CACF. The high degree of nonlinearity in both the process and measurement models of this problem makes it a difficult state estimation problem. This stochastic system is expressed as

|  |  |
| --- | --- |
|  | (17) |

where  is a random variable modelling the process noise sequence. Moreover, the parameters w and  are respectively set as 0.04 and 0.5, as in [10, 11, 16]. Furthermore, the measurement model can be written as

|  |  |
| --- | --- |
|  | (18) |

where *vk* denotes the measurement noise and the parameters  and  are equal to 0.2 and 0.5, respectively. The initial state and the initial covariance of the estimation are taken as x0=1 and P0=2, respectively. Moreover, 100 particles are considered in the EPF and the UPF. The tuned parameters of the ECACF is presented in Table 1.

To investigate the performance of the ECACF, two experiments are performed. In the first experiment, the process noise is non-Gaussian and in the second experiment, the measurement noise is non-Gaussian. The noise properties of these experiments are given in Table 2. The experiments are repeated 500 times for each run to investigate the effect of the random operators of the algorithm and to obtain the average performance.

**Table 1.** The parameters of the ECACF.

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| N | 10 | Number of ants |
| imax | 5 | Maximum number of iterations per step time |
| Nt | 40% | Percent of top ants |

**Table 2.** The noise model of the benchmark problems.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Experiment Number | Process noise | | | Measurement noise | | |
| Distribution | Parameter | Value | Distribution | Parameter | Value |
| 1 [10, 11] | Gamma | Shape | 3 | Gaussian | Mean | 0 |
| Scale | 2 | Variance | 1×10−5 |
| 2 ‎[16] | Gaussian | Mean | 0 | Gamma | Shape | 7 |
| Variance | 1×10−5 | Scale | 2 |

The mean and the variance of the Root Mean Square Error (RMSE), obtained for 500 random runs, are presented in Table 3. It can be observed that the ECACF produces comparable and even better results than other filters in both cases. In addition, the variance of the ECACF is almost the smallest compared to other methods, implying that it has a stable performance according to ‎[25]. Moreover, the box plots and histograms, shown in Fig. 2, compare the RMSE of all filters. The cross line in each box plot indicates the median of the RMSE. The superiority of the ECACF is observed.

**Table 3.** Comparison of mean and variance of the RMSE obtained for 500 independent runs.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Filter | RMSE for Experiment 1 | |  | RMSE for Experiment 2 | |
| Mean | Variance |  | Mean | Variance |
| EPF | 1.0326 | 1.0689 |  | 0.3463 | 0.0035 |
| UPF | 0.7525 | 0.4726 |  | 0.2816 | 0.0014 |
| CACF | 1.0861 | 0.0668 |  | 0.2813 | 0.0016 |
| ECACF | 0.3154 | 0.1142 |  | 0.2019 | 1.107e-6 |

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |
|  |  |
| (c) | (d) |

**Fig. 3.** Comparison of the RMSE obtained for 500 independent runs: (a) and (b) Experiment 1 (non-Gaussian process noise). (c) and (d) Experiment 2 (non-Gaussian measurement noise).

The performance of the ECACF in a single run of the first experiment (non-Gaussian process noise) is shown in Fig. 3. Fig. 3(a) shows the true and estimated state and Fig. 3(b) represents the estimation error. These figures indicate that the proposed algorithm can properly estimate the current state of the nonlinear system in the presence of non-Gaussian process noise. Fig. 4 compares the performance of the ECACF to other filters and shows that the output of the ECACF can track the true output better than other heuristic filters. Moreover, it is observed that the hybridization of the CACF with the EKF improves the accuracy of the CACF. The results, obtained for the second experiment (non-Gaussian measurement noise), are shown in Fig. 5 and Fig. 6. Again, the ECACF shows a successful performance in the presence of non-Gaussian measurement noise.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**Fig. 4.** Performance of the ECACF in the presence of non-Gaussian process noise: (a) True versus estimated state. (b) Estimation error.



**Fig. 5.** Comparison of the ECACF with other filters: Estimation of the state in the presence of non-Gaussian process noise.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**Fig. 6.** Performance of the ECACF in the presence of non-Gaussian measurement noise: (a) True versus estimated state. (b) Estimation error.



**Fig. 7.** Comparison of the ECACF with other filters: Estimation of the state in the presence of non-Gaussian measurement noise.

* 1. Wind Estimation Problem: Processor in the Loop Simulation

Flight of UAVs is often associated with environmental disturbances such as the wind shear. The influence of wind shear on the UAV motion is important particularly during landing and take-off. In this section, the ECACF is utilized to simultaneously estimate the wind shear velocity as well as the current states of the UAV during the flight. Fig. 7 illustrates the block diagram of the estimation problem. In the dynamic modeling of the UAV, it is assumed that the vehicle is rigid. Moreover, sensors have no dynamic while their outputs are corrupted by noise. The measurements are provided by a pitot tube, an Attitude and Heading Reference System (AHRS) and an altimeter. ECACF uses noisy measurements to estimate the wind velocity and the states of the UAV.

The longitudinal dynamic equations of the UAV are stated as follows ‎[26]:

|  |  |
| --- | --- |
|  | (19) |
|  | (20) |
|  | (21) |
|  | (22) |
|  | (23) |

In the above equations, *u* and *w* represent the longitudinal and vertical speeds (ft/sec) and *q*, *θ*, and *h* are pitch rate (deg/sec), pitch angle (deg) and altitude (ft), respectively. Moreover, *δ*e and *δ*t are the elevator deflection (deg) and the throttle setting. Also,U0, γ0 and grepresent the nominal speed (ft/sec), the flight path angle (deg) and the gravitational acceleration (ft/sec2), respectively. X*i*, Z*i* and M*i* (*i*=*u*, *w*, *q*, e, t) are the stability and control coefficients. These parameters are presented in Table 4. The nominal flight speed is U0=235 ft/sec and the initial flight path angle is γ0=-3 deg.

*q*m

Wind Shear Model

*u*m

*θ*m

UAV

*h*m





Altimeter

Pitot tube













ECACF

AHRS



**Fig. 8.** Block diagram of the wind estimation using ECACF.

**Table 4.** Longitudinal Stability and Control Coefficients ‎[26].

|  |  |  |
| --- | --- | --- |
| Parameter | Unit | Value |
| Xu | 1/sec | -0.038  -0.0513  0.00152  0.00005  0.158  0.313  -0.605  -0.041  -0.146 |
| Xw | 1/sec |
| Xq | ft/rad/sec |
| Xe | ft/rad/sec2 |
| Xt | ft/rad/sec2 |
| Zu | 1/sec |
| Zw | 1/sec |
| Zq | ft/rad/sec |
| Ze | ft/rad/sec2 |
| Zt | ft/rad/sec2 | 0.031 |
| Mu | rad/ft/sec | -0.0211  0.157  -0.612  0.459  0.0543 |
| Mw | rad/ft/sec |
| Mq | 1/sec |
| Me | 1/sec2 |
| Mt | 1/sec2 |

Moreover, *u*g and *w*g are the horizontal and vertical components of the wind shear (ft/sec), respectively. In the wind shear model, the variation of the mean wind along the flight path is known. In low altitude, the mean velocity of the wind along the longitudinal axis is assumed as ‎[27]

|  |  |
| --- | --- |
|  | (24) |

where w20 is the measured wind speed at h=20 feet and z0 is a constant the value of which is 0.15 ft for category C (terminal) flight phases such as takeoff, approach, and landing ‎[27] and 2 ft for all other flight phases. The mean wind speed in the flat Earth coordinate frame is transferred to the body frame using the Direction Cosine Matrix.

In the next step, to have a stochastic model of the UAV for the estimation purpose, the deterministic model, given by Eqs. (19) to (23), is augmented by process noises. By defining , , , , and , the stochastic model of the UAV in state space can be expressed as ‎[28]

|  |  |
| --- | --- |
|  | (25) |
|  | (26) |
|  | (27) |
|  | (28) |
|  | (29) |

Moreover, according to Eq. (24) and by defining the augmented state , the horizontal and vertical components of the wind shear in state space can be written as

|  |  |
| --- | --- |
|  | (30) |
|  | (31) |

The dynamic model of the new state is written as

|  |  |
| --- | --- |
|  | (32) |

In Eqs. (25) to (29) and Eq. (32), *ωi* (*i*=1,…,6) are zero mean Gaussian random variables representing the process noise. The measurement model is written as

|  |  |
| --- | --- |
|  | (33) |
|  | (34) |
|  | (35) |
|  | (36) |

where are zero mean white noises, corrupting the measurement.

To investigate the performance of the ECACF in the estimation of the UAV states and the wind components, two experiments are performed. First, a numerical simulation of the system and the estimator is performed. Then, the ECACF is evaluated through a PIL simulation. PIL is a common approach for the verification of compiled object code on the target platform to ensure functional equivalence of code running on the target processor relative to the model behavior captured in simulation ‎[29].

The performance of the ECACF in estimating the horizontal and vertical components of the wind shear is shown in Fig. 8 and Fig. 9, respectively. Moreover, Fig. 10 compares the true and the estimated wind speed at h= 20 ft. The estimated parameter converges to the true parameter, i.e. 20 ft/sec. The performance of the ECACF in estimating the states of the UAV is shown in Fig. 11. These figures show that the proposed filter can properly estimate the states of the UAV and the parameter of the wind shear.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**Fig. 9.** Estimating the horizontal component of the wind shear: (a) True versus the estimated value. (b) Estimation error.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**Fig. 10.** Estimating the vertical component of the wind shear: (a) True versus the estimated value. (b) Estimation error.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**Fig. 11.** The estimated value of the wind speed at h= 20 ft when a wind shear is applied: (a) True versus the estimated value. (b) Estimation error.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |
|  |  |
| (c) | (d) |
|  |  |
| (e) | (f) |
|  |  |
| (g) | (h) |
|  |  |
| (i) | (j) |

**Fig. 12.** Estimation of the states when a wind shear is applied: (a) and (b) longitudinal speed. (c) and (d) normal speed. (e) and (f) pitch rate. (g) and (h) pitch angle. (i) and (j) height.

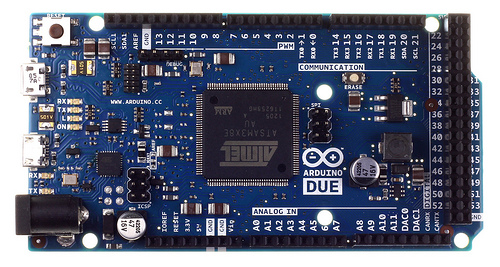
The PIL simulation ‎[30] is performed to verify the ECACF code on the actual embedded processor connected to the simulated model of the UAV in the host computer. The initial conditions and the parameters of the filter are considered the same as those in the software simulation. The general schematic of the PIL Simulation is shown in Fig. 12.

The host computer is connected to the hardware device to send the generated outputs of the UAV simulation model to the hardware through a serial link for each time step. While receiving the output of the UAV model, the hardware device estimates the wind components and the states of the UAV and sends these data to the host computer for data logging and signal monitoring. The results of the PIL simulation are shown in Fig. 13. Fig. 13(a) and Fig. 13(b) compare the simulation and the implementation results in estimating the wind shear components. These results show that the ECACF has been implemented successfully on the processing device.

Simulated model of UAV

The ECACF algorithm







**x̂**



**Fig. 13.** PIL Simulation utilized to verify the ECACF.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**Fig. 14.** Comparison of the PIL and the numerical simulations in the estimation of the wind shear components: (a) Horizontal wind speed. (b) Vertical wind speed.

* 1. Real-time Attitude Estimation of a Quadrotor

In this section, the ECACF is implemented on an experimental setup of a quadrotor, shown in Fig. 14. The quadrotor is free to rotate around its roll, pitch and yaw axes. The angular velocities along three orthogonal axes are measured. ECACF utilizes these noisy measurements for real-time estimation of the angular velocities and the Euler angles. Fig. 15 shows the block diagram of the estimation process.



**Fig. 15**. 3DOF experimental setup of the quadrotor.

Microcontroller

ECACF



3DOF setup of the quadrotor

Measurement Unit

Attitude Command







*p*m

Roll Loop Controller

Pitch Loop Controller

Yaw Loop Controller Control







*u*1

*u*2

*u*3

Microcontroller



*q*m

*r*m

**Fig. 16.** Block diagram of the estimation process.

The schematic of a quadrotor is given in Fig. 16. Each rotor may be considered as a rigid disk rotating around the axis zB in the body fixed frame with an angular velocity . Rotors 1 and 3 rotate in the same directions (e.g. counterclockwise), while rotors 2 and 4 rotate in the opposite direction (e.g. clockwise) to cancel the yawing moment of the whole system.

M4

xB

yB

zB







ZI

XI

xB

yB









M1

M3

M2

dcg

YI

**Fig. 17.** Configuration of a quadrotor and the conventions.

The dynamic model of the quadrotor, obtained from the Newton-Euler method, is stated as follows [31, 32]:

|  |  |
| --- | --- |
|  | (37) |
|  | (38) |
|  | (39) |

In the above equations, (*p*, *q*, *r*) are the angular velocities and (*φ*, *θ*, *ψ*) are roll, pitch and yaw angles. The relationship between Euler angle rates and the body angular rates can be denoted as follows ‎[33]:

|  |  |
| --- | --- |
|  | (40) |
|  | (41) |
|  | (42) |

Moreover, Ixx, Iyy, and Izzare the principle moments of inertia, and Irotor is the inertia of a rotor about its axis. Also, is called the overall residual propeller angular velocity ‎[31], defined as

|  |  |
| --- | --- |
|  | (43) |

The control inputs *u*1, *u*2 and *u*3 are roll, pitch and yaw moments, generated by the propellers, defined as ‎[31]

|  |  |
| --- | --- |
|  | (44) |
|  | (45) |
|  | (46) |

Also, b and d are thrust and drag coefficients, respectively. Moreover, as it is shown in Fig. 16, dcg is the horizontal distance of each rotor from the center of gravity (CG). Therefore, according to Eqs. (44) to (46), the angular velocity commands can be calculated as

|  |  |
| --- | --- |
|  | (47) |
|  | (48) |
|  | (49) |
|  | (50) |

where  is the average angular velocity of the rotors. Parameters of the quadrotor are presented in Table 5. In the next step, to have a stochastic model of the experimental quadrotor for the estimation purpose, the deterministic model, given by Eqs. (37) to (42), is augmented by process noises. By defining , , , ,  and  the stochastic model of the experimental setup in state space representation can be expressed as

|  |  |
| --- | --- |
|  | (51) |
|  | (52) |
|  | (53) |
|  | (54) |
|  | (55) |
|  | (56) |

where *ωi* (*i*=1,…,6) are zero mean Gaussian random variables representing the process noise. The measurement model can be written as

|  |  |
| --- | --- |
|  | (57) |
|  | (58) |
|  | (59) |

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Unit | Value | Description |
| dcg | m | 0.2 | Horizontal distance of each rotor from CG |
| b | Kg.m2 |  | Thrust factor |
| d | Kg.m2 |  | Drag factor |
| Ixx | Kg.m2 | 0.028 | Moment of inertia around *x* axis |
| Iyy | Kg.m2 | 0.031 | Moment of inertia around *y* axis |
| Izz | Kg.m2 | 0.044 | Moment of inertia around *z* axis |
| Irotor | Kg.m2 |  | Moment of inertia of a rotor around the axis of rotation |
|  | RPM | 2000 | Average speed of the engines |

**Table 5.** Parameters of the quadrotor.

where are zero mean white noises, corrupting the measurements. Here, the performance of ECACF in real-time attitude estimation of the quadrotor is investigated and the results are shown in Fig. 17. The results are compared with those of an available AHRS to verify the estimation results. Fig. 17(a) and Fig. 17(b) show the performance of ECACF in estimating roll angle and its rate. Fig. 17(a) represents the measured and the estimated roll rate and Fig. 17(b) represents the estimated roll angle and that obtained from the AHRS. The same results for pitch and yaw channels have been represented in Fig. 17(e) to Fig. 17(f), respectively. These results show the capability of ECACF for a real-time implementation.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |
|  |  |
| (c) | (d) |
|  |  |
| (e) | (f) |

**Fig. 18.** Real-time attitude estimation using ECACF: (a) roll rate. (b) roll angle. (c) pitch rate. (d) pitch angle. (e) yaw rate. (f) yaw angle.

1. Conclusion

In this paper, a hybridization of the CACF and the EKF was performed and a new heuristic filter, called ECACF, was proposed for state estimation of the nonlinear systems. In this filter, a colony of virtual ants, walking through the state space, is utilized to find and track the best state estimation, while the position of ants at the measurement time is updated using the EKF. The pheromone, accumulated gradually around the true states of the nonlinear system, attracts the ants toward the best estimation. In order to investigate the performance of the proposed filter, three problems are investigated: a nonlinear benchmark problem; a multi-dimensional engineering problem and a real-time implementation problem.

First, a nonlinear system was employed to investigate the performance of the new filter in terms of convergence and accuracy. The results showed that the ECACF is able to provide competitive and even better results as compared to CACF and the well-known hybrid filters like the EPF and the UPF. It was observed that the accuracy of the CACF is improved by the hybridization with the EKF. The results also showed the convergence of the ECACF in this problem.

Second, an engineering problem including the estimation of the UAV states and the wind velocity components was employed using the ECACF. The results showed that the ECACF has high performance for the state estimation of a system with more than one state. Moreover, the small estimation error proved the successful performance of the ECACF in a nonlinear engineering problem. Additionally, the ECACF was successfully implemented on a target platform for carrying out a PIL simulation.

In the third problem, the ECACF was applied to the state estimation of a 3DOF experimental quadrotor to investigate its effectiveness in practice. In this regard, the states including orientation and angular rates were estimated in a real-time. The results showed that the proposed algorithm can effectively be implemented in a real-time application.

**References**

1. B. Ristic, S. Arulampalam, N. J. Gordon, Beyond the Kalman filter: Particle filters for tracking applications, Artech house, 2004.
2. H. Nobahari, A. Sharifi, H. MohammadKarimi, Swarm intelligence techniques applied to nonlinear systems state estimation, in: Advances in Heuristic Signal Processing and Applications, Springer, 2013, pp. 219-241. doi:10.1007/978-3-642-37880-5 10.
3. H. Nobahari, A. Sharifi, A novel heuristic filter based on ant colony optimization for non-linear systems state estimation, in: Computational Intelligence and Intelligent Systems, Springer, 2012, pp. 20-29. doi: 10.1007/978-3-642-34289-9\_3.
4. E. A. Wan, R. Van Der Merwe, The unscented kalman filter for nonlinear estimation, in: Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000, Ieee, 2000, pp. 153-158.
5. M. S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking, IEEE Transactions on signal processing 50 (2) (2002) 174-188. doi:10.1109/78.978374.
6. D. Simon, Optimal state estimation: Kalman, H infinity, and nonlinear approaches, John Wiley & Sons, 2006.
7. G. Tong, Z. Fang, X. Xu, A particle swarm optimized particle filter for nonlinear system state estimation, in: Evolutionary Computation, 2006. CEC 2006. IEEE Congress On, IEEE, 2006, pp. 438-442. doi:10.1109/cec.2006.1688342.
8. N. J. Gordon, D. J. Salmond, A. F. Smith, Novel approach to nonlinear/non-gaussian bayesian state estimation, in: IEE Proceedings F (Radar and Signal Processing), Vol. 140, IET, 1993, pp. 107-113. doi:10.1049/ip-f-2.1993.0015.
9. A. Doucet, N. De Freitas, N. Gordon, Sequential monte carlo methods in practice. series statistics for engineering and information science (2001).
10. L. Liang-Qun, J. Hong-Bing, L. Jun-Hui, The iterated extended Kalman particle filter, in: Communications and Information Technology, 2005. ISCIT 2005. IEEE International Symposium on, Vol. 2, IEEE, 2005, pp. 1213-1216. doi:10.1109/iscit.2005.1567087.
11. R. Van Der Merwe, A. Doucet, N. De Freitas, E. A. Wan, The unscented particle filter, in: Advances in neural information processing systems, 2001, pp. 584-590.
12. T. Higuchi, Monte carlo filter using the genetic algorithm operators, Journal of Statistical Computation and Simulation 59 (1) (1997) 1-23. doi:10.1080/00949659708811843.
13. T. Clapp, Statistical methods for the processing of communications data, Ph.D. thesis, University of Cambridge (2001).
14. P. Torma, C. Szepesva'ri, Ls-n-ips: an improvement of particle filters by means of local search, IFAC Proceedings Volumes 34 (6) (2001) 687-691. doi:10.1016/s1474-6670(17)35258-8.
15. L. Xiaolong, F. Jinfu, L. Qian, L. Taorong, L. Bingjie, A swarm intelligence optimization for particle filter, in: Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on, IEEE, 2008, pp. 1986-1991.
16. J. Zhong, Y.-f. Fung, M. Dai, A biologically inspired improvement strategy for particle filter: Ant colony optimization assisted particle filter, International Journal of Control, Automation and Systems 8 (3) (2010) 519-526. doi:10.1007/s12555-010-0304-7.
17. H. Han, Y.-S. Ding, K.-R. Hao, X. Liang, An evolutionary particle filter with the immune genetic algorithm for intelligent video target tracking, Computers & Mathematics with Applications 62 (7) (2011) 2685-2695.
18. M. Ahmadi, H. Mojallali, R. Izadi-Zamanabadi, State estimation of nonlinear stochastic systems using a novel meta-heuristic particle filter, Swarm and Evolutionary Computation 4 (2012) 44-53. doi:10.1016/j.swevo. 2011.11.004.
19. M. Kiani, S. H. Pourtakdoust, State estimation of nonlinear dynamic systems using weighted variance-based adaptive particle swarm optimization, Applied Soft Computing 34 (2015) 1-17. doi:10.1016/j.asoc.2015.04. 029.
20. H. Nobahari, S. M. Zandavi, H. Mohammadkarimi, Simplex filter: A novel heuristic filter for nonlinear systems state estimation, Applied Soft Computing 49 (2016) 474-484. doi:10.1016/j.asoc.2016.08.008.
21. H. Nobahari, M. Raoufi, A. Sharifi, A heuristic filter based on fire-fly algorithm for nonlinear state estimation, in: Computational Intelligence (SSCI), 2016 IEEE Symposium Series on, IEEE, 2016, pp. 1-6. doi:10.1109/ssci.2016.7850275.
22. S. Pourtakdoust, H. Nobahari, An extension of ant colony system to continuous optimization problems, Ant colony optimization and swarm intelligence (2004) 158-173. doi:10.1007/978-3-540-28646-2\_27.
23. I. Fister, M. Perc, S. M. Kamal, A review of chaos-based firefly algorithms: perspectives and research challenges, Applied Mathematics and Computation 252 (2015) 155-165. doi:10.1016/j.amc.2014.12.006.
24. S. J. Julier, J. K. Uhlmann, A new extension of the kalman filter to nonlinear systems, in: Int. symp. aerospace/defense sensing, simul. and controls, Vol. 3, Orlando, FL, 1997, pp. 182{193. doi:10.1117/12.280797.
25. R. Roll, A mean/variance analysis of tracking error, The Journal of Portfolio Management 18 (4) (1992) 13-22. doi:10.3905/jpm.1992.701922.
26. H.-S. Ju, C.-C. Tsai, Longitudinal auto-landing controller design via adaptive backstepping, International Journal of Adaptive Control and Signal Processing 23 (7) (2009) 640-666. doi:10.1002/acs.1055.
27. R. Vepa, Flight Dynamics, Simulation, and Control: For Rigid and Flexible Aircraft, CRC Press, 2014.
28. A. Sharifi, H. Nobahari, Multiple model filters applied to wind model estimation for a fixed wing uav, in: Mechanical and Aerospace Engineering (ICMAE), 2016 7th International Conference on, IEEE, 2016, pp. 109-115. doi:10.1109/icmae.2016.7549518.
29. G. Gegic, In-the-loop testing aids embedded system validation (2009).
30. R. Antão, A. Mota, R. E. Martins, J. T. Machado, Processor-in-the-loop simulation, in: Type-2 Fuzzy Logic, Springer, 2017, pp. 111-124.
31. S. Bouabdallah, R. Siegwart, Full control of a quadrotor, in: Intelligent robots and systems, 2007. IROS 2007. IEEE/RSJ international conference on, Ieee, 2007, pp. 153-158. doi:10.1109/iros.2007.4399042.
32. H. Nobahari, A. Sharifi, Continuous ant colony filter applied to online estimation and compensation of ground effect in automatic landing of quadrotor, Engineering Applications of Artificial Intelligence 32 (2014) 100-111. doi:10.1016/j.engappai.2014.03.004.
33. P. H. Zipfel, Modeling and simulation of aerospace vehicle dynamics, reston, va: American institute of aeronautics and astronautics (2007).