Neural-Network for Position Estimation of a Cable-Suspended Payload Using Inertial Quadrotor Sensing

Julien Mellet, Jonathan Cacace, Fabio Ruggiero and Vincenzo Lippiello
PRISMA Lab, Department of Electrical Engineering and Information Technology, University of Naples Federico II, Via Claudio 21, 80125, Naples, Italy

Keywords: Neural Network, State Estimation, Quadrotor, Cable Suspended Payload.

Abstract: This paper considers a standard quadrotor drone with a cable-suspended payload and minimal sensor configuration. A neural network estimator is proposed to perform accurate real-time payload position estimation. A novel proprioceptive feedback measurement method is proposed, and a neural network has been trained with domain randomization. The network shows accurate zero-shot estimation, even with excitations never seen by the system before. This preliminary work has been tested in a simulated environment and aims to show that only onboard inertial sensing is enough to achieve the sought task. The presented work may open new applications for drone transportation in real environments subject to several perturbations.

1 INTRODUCTION

For aerial vehicles, there is a weight obsession such that, especially for transportation tasks, each saved gram on the platform is a gram for the payload (Anderson and Gaston, 2013). Thanks to its simplicity, a cable-suspended payload under an aerial vehicle remains an elegant way to move any object through the air (see Fig. 1). Compared to a mechatronic arm, this system is lightweight and easy to install, utilizing a single passive cable fixation point. (Suarez et al., 2020). Despite new aerial platforms like omnidirectional drones, quadrotors have demonstrated a decade-long reliability (Bodie et al., 2019). A quadrotor with suspended load is comparable to helicopters achieving object transport but with a difference in agility and sensory system (Wendel et al., 2006). Nevertheless, research tends to increase flight accuracy by embedding as many sensors as possible at the expense of lightness and agility (Lanegger et al., 2006), (Panetsos et al., 2022).

This preliminary study presents an approach that simplifies suspended payload position estimation in a standard quadrotor, eliminating the need for exteroceptive sensor processing. Hence, only internal state configuration at a high temporal rate is employed instead of cameras or any position sensor. Such systems are robust to environmental variations. The disturbances induced by the payload on the system are used to let a specially trained network find a correlation with the payload position. To the best of the authors’ knowledge, no prior work achieved such accuracy in estimating suspended payload position in minimal aerial transportation setups.

1.1 Related Works

Conventional approaches to cable-suspended payload transportation use external sensors such as cam-
era (Tang et al., 2018), (Guo and Leang, 2020). Getting a direct measurement of the payload position is a reasonable option. Even with a relatively low measurement rate compared to the low-level controller, it allows making aggressive flights (Tang et al., 2018).

However, cameras suffer from several problems like brightness (close to shadow), flickering (while flying over the sea), or fogging (in a humid area), making their use standalone unsafe for reliable industrial applications. The issue has also been revealed by (Lee and Kim, 2017), where the proposed solution consists of adding force sensors to the system. Equivalent sensing solutions have been adopted by (Lv et al., 2021), with the addition of a universal joint between the drone and the cable for the second one. Recently, (Panetsos et al., 2022) used four different sensors to get accurate cable state estimation, while (Outeiro et al., 2023) proposed an adaptive geometric control method with asymptotic tracking stability.

A similar approach to ours is described by (Kaufmann et al., 2020) and (Cioffi et al., 2022) to perform agile maneuvers. The use of internal sensing in those papers is called sensorimotor, but for the unification of the terms in robotics, we will call it proprioception, like it has been done in (Lee et al., 2020). Even if we do not have access to direct measurement of the motor speed, the PWM (Pulse-Width Modulation) command gives a fair proportional estimation. A neural network (NN) is implemented to control the drone. In particular, the work done by (Cioffi et al., 2022) learns inertial odometry and gets accurate position estimation without using any visual perception. However, the positioning tracking has been done with previously known trajectories. In contrast, we implemented a neural network trained by supervised learning on the estimation stage of the tethered payload controller.

One of the most advanced research on payload state estimation using quadrotor proprioception is for parameter estimation (Prkačin et al., 2021). Recently, such a work has been improved by (Prkačin et al., 2021) with the implementation of an extended Kalman filter (EKF). However, estimation of load parameters remains challenging with the employed fast Fourier transform technique, getting only off-line results. Unfortunately, even if the research looks in an interesting direction with a minimal drone setup, real-time performance has not been reached. Real-world experiments showed poor results in tracking the angles of the payload. The system is non-linear, and the IMU is noisy while drones fly at high velocity. This makes classical EKF implementation unusable for real flights application. To tackle real-time, we use the concept of a neural observer (Chen et al., 2018) with a recurrent NN to access indirectly measured data (Habtom and Litz, 1997).

1.2 Contribution
In this paper, we prove the feasibility of real-time position estimation of a cable-suspended end-effector using only inertia sensors onboard a standard quadrrotor. Contrary to the classic EKF approach, our method does not need to define any parameter a priori. We used a supervised learned network, making the position estimation of the suspended load for a standard quadrotor attitude controller. The neural estimator is trained in a simulated environment, with domain randomization, and runs the software controller in the loop. We reached zero-shot generalization of the network for load position estimation. After being trained on our dataset, the network has the capability to predict previously unseen perturbations, in particular, direct injection of energy into the end-effector. This shows the consistency of the estimator implementation, as well as its generalization capabilities.

2 MATERIALS AND METHODS
Making payload pose estimation is not a trivial task, and the classical filter approach showed accuracy limitations (Prkačin et al., 2021). In this section, we first define our model, which is needed for the simulation, and implement an attitude controller. Then, we focus on data processing to measure our features to train the network. Finally, to smooth the training, we gradually increase the measurement domain exploration with progressively more sophisticated trajectories.

2.1 Model
For the quadrotor, we consider the dynamic model with the Euler angles (Ollero and Siciliano, 2019). Here we use FLU (Front-Left-Up) convention to define the body frame $B$ with axes $\{x, y, z\}$, and the world frame $W$ with axes $\{x_w, y_w, z_w\}$. We define the drone position and its attitude as $p_b = [x_b, y_b, z_b] \in \mathbb{R}^3$ and $\eta_b = [\theta_b, \phi_b, \psi_b] \in \mathbb{R}^3$, respectively, with their time derivatives $p_B$ and $\eta_B$. The attitude can also be defined with $R_b \in SO(3)$, the rotation matrix from $B$ to $W$, the special orthogonal group of dimension three, from which the roll-pitch-yaw angles $\phi_b$, $\theta_b$, and $\psi_b$, respectively, can be extracted. These give the linear and angular accelerations of the base as,

$$\ddot{p}_b = g e_3 + \frac{1}{m} u_T R_b e_3,$$ (1)
Payload position controller
Payload attitude controller
Quadrotor attitude controller
Plant controller
Neural estimator

Figure 2: Control layout architecture with neural network state estimation.

Figure 3: Schematic representation of the quadrotor (in blue) with the cable-suspended payload (in red). The terms \( l_i, i \in \mathbb{R} \), with \( l_i \gg l_c \) are the support length and the cable length, respectively. Besides, \( \theta_i \in \mathbb{R}, i = 1, \ldots, 4 \) are the rotation speed of the motors, while \( \theta_1, \theta_2 \in \mathbb{R} \) are the angles of the cable with respect to the drone.

\[
\tilde{\eta}_b = M^{-1}(\eta_b) - C(\eta_b, \eta_b) + Q^T(\eta_b) \tau_u, \quad (2)
\]

where \( g \in \mathbb{R} \) is the gravity, \( m \in \mathbb{R} \) is the drone mass, and \( e_3 = [0, 0, 1]^\top \). Moreover, \( M(\eta_b) = Q(\eta_b)^T I_b Q(\eta_b) \in \mathbb{R}^{3 \times 3} \) is the symmetric and positive definite (provided that \( \theta \neq \frac{\pi}{2} \)) mass matrix, \( I_b \in \mathbb{R}^{3 \times 3} \) is the drone inertia matrix, and \( Q(\eta_b) \in \mathbb{R}^{3 \times 3} \) is the transformation matrix such that \( \omega_b = Q(\eta_b) \omega_b \), where \( \omega_b \in \mathbb{R}^3 \) is the angular velocity of the B with respect to the W expressed in B. Finally, \( C(\eta_b, \eta_b) = Q^T(\eta_b) S(\eta_b) Q(\eta_b) \in \mathbb{R}^{3 \times 3} \) is the Coriolis matrix, with \( S(\cdot) \in \mathbb{R}^{3 \times 3} \) the skew-symmetric operator, \( \tau_\omega = [\tau_x, \tau_y, \tau_z]^\top \in \mathbb{R}^3 \) is the torque control vector, and \( \tau_u \in \mathbb{R}^+ \) is the total thrust. The physical control inputs to the system, that is the propeller velocities \( \omega_i \in \mathbb{R} \), with \( i = 1, \ldots, 4 \) (see Fig. 3), can be retrieved from the torques and the total thrust through the allocation matrix as follows,

\[
\begin{bmatrix}
\tau_u \\
\tau_\omega \\
\end{bmatrix} =
\begin{bmatrix}
C_T & C_T & C_T & C_T \\
0 & -C_T & 0 & C_T \\
-C_Q & C_Q & C_Q & -C_Q \\
\end{bmatrix}
\begin{bmatrix}
\omega_1^T \\
\omega_2^T \\
\omega_3^T \\
\omega_4^T \\
\end{bmatrix}, \quad (3)
\]

where \( C_T, C_Q \in \mathbb{R}^+ \) are the thrust constant and drag factor, respectively.

For the suspended-cable, several models were proposed in the literature, such as finite element approximation (Goodarzi et al., 2014), elastic rope (Kotaru et al., 2017), or a rigid bar (Tang et al., 2018). The last model has been chosen for ease of simulation. It represents the most interesting case for transporta. Details of the model are presented by (Sreenath et al., 2013) and explained in Fig. 3. Recall that the quadrotor is a differentially flat system (Rathnam et al., 1995), while the quadrotor with a cable-suspended payload is a differentially flat hybrid system (Tang et al., 2018; Sreenath et al., 2013). Hence, all state and input variables are defined through non-linear equations involving flat variables and their respective derivatives.

The load is considered as a point mass, the mass of the cable is neglected, and the cable is considered rigid. The variables are defined in Fig. 3. The position of the quadrotor in W can be retrieved from the load position as

\[
p_b = q_l - l_c p_t, \quad (4)
\]

where \( q_l \in \mathbb{R}^3 \) is the load position in W and \( p_t \in \mathbb{R}^2 \), with \( \mathbb{R}^2 \) the manifold of unit vectors in \( \mathbb{R}^3 \), is the unit vector pointing the load from the quadrotor’s center of mass.

\subsection*{2.2 Control}

The controller presented by (Sreenath et al., 2013) and applied by (Tang et al., 2018) is here briefly reported. The state and input are defined as, \( x = [p_l, q_l, \dot{p}_b, \dot{q}_b, \omega_b^\top, \omega_b^\top]^\top \) and \( u = [\tau_u^\top, \tau_\omega^\top]^\top \), respectively.

First, define the low level quadrotor attitude controller as

\[
\tau_b = -K_R e_R - K_o e_\omega + S(\omega_b^\top) I_b \omega_b^\top - I_b S(\omega_b^\top) R_{b,d}^\top \omega_{b,d}^\top - R_{b,d}^\top R_{b,d} \omega_{b,d}^\top, \quad (5)
\]

where \( K_R, K_o \in \mathbb{R}^{3 \times 3} \) are diagonal gain matrices, \( e_R = \frac{1}{2} (R_{b,d}^\top R_{b,d} - R_{b,d}^\top R_{b,d}) \) is the quadrotor attitude error, \( e_\omega = \omega_b^\top - R_{b,d}^\top \omega_{b,d} \in \mathbb{R}^3 \) is the angular velocity error, and \( \vee \) is the vee operator mapping a skew-symmetric matrix into the vector that generated it, \( R_{b,d} \in SO(3) \) is the desired quadrotor rotation matrix, \( \omega_{b,d} \in \mathbb{R}^3 \) is the desired angular velocity expressed in the desired body frame.

The desired \( R_{b,d} \) and \( \omega_{b,d} \) are retrieved from the position controller as explained by (Tang et al., 2018).
The thrust input results from,

\[ f_b = u_T \rho_b e_3, \]

where,

\[ u_T = K_p e_p + K_p e_p + K_i \int e_p dt, \]

with \( K_p, K_p, K_t \in \mathbb{R}^3 \) some diagonal and positive definite gain matrices, \( e_p = S(p l) \rho \eta \) the error function with \( \rho \eta, \rho \eta \in \mathbb{R}^3 \) the desired load position in \( \mathbb{R}^3 \), and \( e_p = p l - S(p l) \rho \eta \rho \eta \) the time derivative. Then, the desired quadrotor attitude is,

\[ R_{d:b} = \begin{bmatrix} S(b)c \\ |S(b)c| \end{bmatrix} b, S(b)c, |S(b)c|, c, \]

with \( || \cdot || \) the Euclidean norm, \( b = \frac{u_T}{||u_T||} \in \mathbb{R}^3 \), and \( c = [\cos(\psi_b), \sin(\psi_b), 0] \in \mathbb{R}^3 \).

Finally the payload position controller is defined as,

\[ q_{l:d} = K_q e_q + K_q e_q + K_{t:i} \int e_q dt, \]

with \( K_q, K_q, K_{t:i} \in \mathbb{R}^3 \) some diagonal and positive definite gain matrices, \( e_q = q_l - q_{l:d} \in \mathbb{R}^3 \) the error function, and \( e_q = q_l - q_{l:d} \in \mathbb{R}^3 \) its time derivative.

The controller implementation is presented in Fig. 2 and the NN estimator is explained in the next section.

### 2.3 Neural Estimation

Through all the data provided by the IMU, the magnetometer has been found to add too much noise and randomness to the result. Then, only the accelerometer and the gyroscope are used.

Compared to (Kaufmann et al., 2020), we found that sampling measurements with synchronization of all the features were getting accurate results. Normalization is done with min-max normalization. Assuming \( X \in \mathbb{R} \) a scalar, we use the unit-based normalization,

\[ X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}. \]

To avoid any overshoot on measurements \( X_{\text{meas}} \in \mathbb{R} \), boundaries limits are fixed with saturation values such that,

\[ X_{\text{meas}} \in [X_{\text{min}}, X_{\text{max}}]. \]

The main network architecture Fig. 4 remains condensed, with relatively few hidden layers. Keeping the network as small as possible is to implement it on a small computational unit and reduce the processing latency. It has three incoming features branches with the IMU \((p_b, \eta_b) \in \mathbb{R}^6\) the motor speeds \( \omega \in \mathbb{R}^4 \), and the control inputs \((u_T, \eta_b, d) \in \mathbb{R}^4\). There is a total of fourteen scalar input features \([p_b, \eta_b, \omega, u_T, \eta_b, d]\) preprocessed as mentioned above.

For a later comparison study, we want to compare different promising NNs, namely, a time convolutional network (TCN) and two recurrent neural networks (RNN). For both cases, networks are trained on a history of \( N \) time past steps to the current measurement that are defined later. Even if TCN proved its efficiency in neural estimation for drones (Kaufmann et al., 2020; Cioffi et al., 2022), RNN looks to be a more appropriate option. These layers take advantage of processing sequential data to make the prediction. In this study, we used Long Short-Term Memory (Hochreiter and Schmidhuber, 1997) (LSTM) hidden neurons that have also been used by (Perigal et al., 2022; Jung et al., 2022). We can notice that simple RNN and gated recurrent unit (GRU) have also been tested during this research, but the results were similar to LSTM with poorer prediction accuracy. For this reason, LSTM is the chosen recurrent net architecture.

The recurrent network trained with \( N = 15 \) timesteps of the past proprioceptive measurements follows. Even if the eight past measures were used by (Kaufmann et al., 2020), we got better performance with fifteen. Then, two fully connected layers of MLPs give the cable angles \((\theta_1, \theta_2)\). We use differentiation to obtain the angular velocity \( \dot{\theta}_1 \) needed for the controller in Fig. 2.
2.4 Training

Compared to (Lee et al., 2020), (Faust et al., 2017) that substantially used reinforcement learning to train their networks, we used supervised learning. However, we kept the principle of accelerating the training with a given policy. Thus, we proposed to begin the training with simple trajectories, letting the loss converge smoothly. Then, we complicated the trajectories to finish with erratic movements. Here, only state estimation is performed by the network. The advantage of virtual training is to have an infinite flight time with variable parameters. The payload position controller is fed with circular, lemniscate, and random manual trajectories (see Fig. 5).

We trained the NN estimator through ten datasets with the five different trajectories, depicted in Fig. 5, and two different payloads of 200 g and 800 g. The sum of the datasets gives a total of \(140 \times 10^3\) timestamped measurements for 50 min flight time. We split each trajectory dataset in two, with 20% for validation and the remaining 80% for training.

To predict the load position, \(p_l\), a time window of size 16 has been used. Thanks to the 15 past measured features, the 16th label is predicted. Because the regressive network has to keep consistency throughout time, no data shuffle was made. Nevertheless, each data set was split up into batches of size 256. Trying different batch sizes, it turned out this last was a good trade-off. For the simulation, we used Gazebo, a simulator with a physics engine (Smith, 2008). The propellers’ drag, lift coefficients and IMU characteristics are simulated through the internal plugin. We simulate our quadcopter with a stick with a two-degree-of-freedom (DoF) universal joint, presented in Fig. 3. Compared to real flights, the simulator does not provide any aerodynamic effects. As a result, the payload is static under the drone while hovering. We then propose to make domain randomization through the excitation of the end-effector. It consists in applying a relatively small bounded random force on the end-effector. In our case, we selected a maximum force of 0.2 N. It is now about comparing the network’s performances.

Table 1: Architectures of the three networks that are compared for the study. LSTM2 with two recurrent layers and an MLP before the output. TCN with a convolutional layer followed by an MLP. LSTM embedding a unique recurrent layer and two MLPs.

<table>
<thead>
<tr>
<th>Hidden layer 1</th>
<th>Hidden layer 2</th>
<th>Hidden layer 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM 256</td>
<td>MLP 128</td>
<td>MLP 64</td>
</tr>
<tr>
<td>LSTM 128</td>
<td>MLP 128</td>
<td>X</td>
</tr>
<tr>
<td>LSTM 128</td>
<td>MLP 128</td>
<td>MLP 64</td>
</tr>
</tbody>
</table>

2.5 Networks Comparisons

The first set of experiments compares the accuracy in the prediction of three promising neural architectures. Table 1 compares the architectures of the three networks used for the research, with two recurrent layers and an MLP before the output. TCN with a convolutional layer followed by an MLP. LSTM embeds a unique recurrent layer and two MLPs. Then, the training through the different trajectories is presented in Fig. 6.

One can notice that the validation loss decreases after each trajectory train. This points out the improvement of the estimation prediction with more and
more sophisticated trajectories. Nevertheless, even if TCN showed a better accuracy on the validation set, RNN performs better with the test set (see Fig. 7).

The better performance of TCN in Fig. 7 is noticeable on the validation set, while LSTM is more accurate on the test set. LSTM2 and even more LSTM has thus a generalization capability. Both temporal convolution and recurrent networks result in the same order of magnitude, thanks to temporal measurements.

The following section uses only LSTM to understand what implies the change in payload mass on the NN estimator.

3 APPLICATIONS

NN estimation takes a certain time to process. We made an inference model of our network to accelerate pose estimation during the experiments. We improved the pose estimation processing from a mean response time of $4 \times 10^{-2}$ s to an instant estimation of $10^{-4}$ s without losing accuracy.

3.1 Response to External Perturbations

Because the weight of the payload directly impacts drone perturbation, and by implication the inertial sensor measurements, we want to understand how the weight impacts pose estimation of the end-effector. To measure the accuracy of prediction, we have perturbed the load on the single $x$-axis with 40 N for the 200 g load (see Fig. 8) and with 100 N for the 800 g load (see Fig. 9).

For both plots, with light-weight (see Fig. 8) and heavy payload (Fig. 9), we notice that the estimation follows the ground truth through time with a drop in accuracy at the perturbation. Moreover, no perturbation is measured on $\theta_1$, the non-perturbed axis, showing the independence of each angle estimation.

Another critical point is the noise reduction in payload state estimation with a heavier load. This phenomenon should be explained for two reasons: firstly, because the heavier load, with greater inertia, have slower dynamics, which is easier to predict; secondly, because the perturbation of the largest mass has more effect on the system and disturbance can be measured with higher intensity by the inertial unit.

3.2 Use Case Example

The proposed estimator was then applied in a pick-and-place scenario. The idea is to understand the estimation performance without training the network for this specific application. The drone picks up a 600 g package and transports it to a basket at the target location. Figure 1 presents the environment in which the system is used. The drone flies at a relatively slow speed, keeping the near-hovering assumption.

Compared to the system subjected to simple disturbances (see Figs. 8-9), we can notice a more significant mean absolute error (MAE) during the entire task in Fig. 10. The same comments as previously can...
Figure 10: Cable orientation estimation during a pick and place task. The payload is grasped at $t = 580$ s, and released at $t = 1310$ s. The mean absolute error (MAE) between ground-truth in green and estimated angle in red, has been measured at 0.08365 rad.

be made, on the influence of weight on estimation. For $\theta_1$, MAE is more significant with the unloaded end-effector, while precision increases with the payload transported. An estimation bias has to be noticed for $\theta_2$ during the carriage. Likewise, the grasping action being similar to a prompt perturbation, the same result is observed as above, with a drop in accuracy. In contrast to the grab, the release is smoother, and the accuracy of the estimate changes from a lower to a higher MAE.

4 CONCLUSION AND FUTURE WORK

This preliminary study demonstrates that inertial measurements are sufficient for estimating payload position in cable-suspended drone systems, enabling their integration into the controller. Moreover, indirect neural pose estimation has been proven accurate in performing stable transportation tasks near hovering. The hardware complexity has thus decreased. This estimator suits cable-suspended underwater transport, free from sun reflection flickering on water waves.

Future work will focus on transferring the NN from simulation to reality thanks to domain randomization presented in this paper. The core idea is to test the neural estimator in challenging-to-simulate environments. On the other hand, we would like to take advantage of this estimator to perform aggressive maneuvers implementing model predictive control.

ACKNOWLEDGEMENTS

The research leading to these results has been supported by the AERAL-CORE project (Horizon 2020 Grant Agreement No. 871479) and by the AEROTRAIN Project, European Union’s Horizon 2020 Research and Innovation Program under the Marie Skłodowska-Curie Grant Agreement 953454. The authors are solely responsible for its content.

REFERENCES


