arXiv:2305.02705v1 [cs.RO] 4 May 2023

时间最优四旋翼飞行的制导与控制网络

Sebastien Origer[1], Christophe De Wagter[1],Robin Ferede[1],Guido C.H.E. de Croon[1],Dario Izzo[2]

摘要—实现快速自主飞行需要计算高效且鲁棒的算法。为此，我们训练制导与控制网络，使其近似从能量最优到时间最优飞行的最优控制策略。我们表明，越接近时间最优的“bang-bang”控制曲线，策略就越难以学习。我们还评估了了解四旋翼无人机最大转子角速度的重要性，并表明高估或低估这一限值会降低飞行的鲁棒性。我们提出一种算法，用于在机载识别当前最大转子角速度；此外，还提出一种网络，使其能够根据识别出的限值调整策略。最后，我们通过学习将连续航路点纳入考量，拓展了此前关于制导与控制网络的研究。我们以与基于微分平坦性的最小急动度基准控制器相近的圈速飞行一条 4×3米 航迹，同时受益于制导与控制网络所具备的灵活性。

索引词— **G&CNET、最优控制、模仿学习、端到端、时间最优**

视频：https://youtu.be/FrwpODT0HKQ



图 1. 使用 Parrot Bebop 1 和制导与控制网络完成最快一圈的飞行路径，该网络通过学习考虑前方两个航点来计算最优控制输入。航迹为一个 4×3m 的矩形，四个航点分别位于每个橙色门的中心。

I. 引言

MICRO 微型飞行器（MAVs）是用途广泛的机器人，在搜救任务、娱乐、电影摄影、配送和检查等领域具有明确的应用价值。它们的尺寸、灵活性、速度以及垂直起降能力，为探索尚未开发的可能性打开了全新的世界。此外，其低成本以及自主飞行所固有的挑战，使其成为推动机器人研究前沿的理想平台。

随着对能够自主执行任务的无人机需求日益增长，实现自主且时间最优的飞行变得至关重要。时间最优飞行十分重要，因为某些应用（如搜救任务）的成败取决于无人机抵达目的地的速度。此外，需要长航程的应用（如海上风力涡轮机巡检）也能从时间最优控制解中受益，因为多旋翼飞行器的最优航程速度通常相对较快 [1]。这对于未配备固定翼的无人机（如四旋翼无人机）尤为重要，因为它们的飞行航程有限。自主且时间最优的飞行已成为许多研究的课题 [2]–[9] ，无人机竞速比赛也包括.....

*本研究得到欧洲航天局的支持[1]航空航天工程学院微型飞行器实验室，代尔夫特理工大学，2629 HS 代尔夫特，荷兰 Sebastien.Origer@outlook.com、C.deWagter@tudelft.nl、R.Ferede@tudelft.nl、G.C.H.E.deCroon@tudelft.nl[2]先进概念团队，欧洲航天局，Keplerlaan1，2201 AZ，诺德韦克，荷兰。Dario.Izzo@esa.int

AlphaPilot 挑战赛 [2]正在组织中，以进一步推动该领域的研究。由于空气动力学效应对于时间最优飞行 [10]变得更加显著，并且无人机受到有限机载计算能力的限制，需要克服的挑战之一是创建稳健且计算高效的控制算法。

传统上，自主飞行问题被分解为三个主要步骤：感知（状态估计）、规划（轨迹生成）和控制（轨迹跟踪）。其中一些步骤可以融合在一起。事实上，人们通常会区分两种控制策略：轨迹跟踪方法和轨迹优化方法。前者首先生成轨迹，然后通过控制器对其进行跟踪；而后者则将这两个步骤结合起来，即直接根据状态计算控制指令。尽管当前最先进的轨迹跟踪方法，例如微分平坦性控制（DFBC）[11]或模型预测轮廓控制（MPCC） [4]，能够实现较高的速度，但通常第一步（轨迹生成）的计算开销过大，无法在无人机 [4]，上进行机载求解，除非采用质点模型等简化模型。这会带来问题，因为扰动不可避免地会导致四旋翼无人机偏离预先计算的轨迹。

机器学习的最新进展在自主飞行的三个主要环节中展现出令人鼓舞的成果 [2]，[8]，[12]，[13]。在问题定义合理、训练数据集足够大、网络架构合适且计算能力充足的情况下，如今人工神经网络能够学习

以一定精度逼近任意函数。制导与控制任务的主要机器学习范式是强化学习（RL） [8] 和模仿学习[13]。强化学习提供了一种框架，使网络能够通过向环境中添加噪声来学习应对不确定性；当动态系统的部分内容仍未建模时，这一方法尤其有用。不过，也可以通过模仿学习来应对未建模效应 [13]。此外，关于行星际转移 [14]–[16] 和四旋翼无人机 [12], [13] 的先前研究表明，网络能够直接从大型最优轨迹数据集中学习最优状态反馈。这便有了制导与控制网络（G&CNET）这一术语。G&CNET 可将状态直接映射为原始控制指令，推理所需的计算成本很低，而且非常灵活，因为无需重新计算最优轨迹。

本文改进了以往关于 G&CNET 的工作 [13] ，以提高四旋翼无人机的飞行速度，并作出四项主要贡献。我们从能量最优控制迈向时间最优控制，并给出了训练 G&CNET 以逼近相应最优控制策略时的损失值。我们提出一种自适应方案，用于估计螺旋桨的最大角速度，并据此调整控制指令，使其保持时间最优。我们开发了一种训练方法，使网络能够考虑后续两个航点的视界，输出时间最优的原始控制指令。我们证明，双航点网络能够应对飞行过程中的动态航点位置。

本文结构如下。首先，介绍四旋翼无人机模型、最优控制问题（OCP）、模仿学习过程和实验设置（第 II 节）。随后，我们研究学习并执行时间最优 OCP 的任务，并展示从能量最优控制转向时间最优控制问题后，控制策略会变得多么难以学习（第 III 节）。飞行越接近时间最优，四旋翼无人机的旋翼饱和程度就越高，即控制策略越接近“开关式”控制。鉴于此，我们评估了在四旋翼无人机飞行过程中旋翼可达到的最大转速限制下训练 G&CNET 的重要性。我们还提出一种算法，可结合自适应 G&CNET 识别当前的最大转速限制

其控制策略会根据已识别的限制进行调整（第IV节）。最后，我们将此前关于 G&CNET 的研究从单个航点扩展至连续航点飞行（第V节），从而提升四旋翼无人机的快速飞行能力。我们还将所得控制器与另一种采用时间最优最小快变轨迹的基准方法进行了比较 [11]。

图 1 显示了使用 G&CNET 在 4×3m 航迹上飞行一圈的情况，该网络经过训练，能够将两个连续航路点纳入考虑。

II. 方法

A. 四旋翼无人机模型

在本研究中，我们采用图 2 所定义的两个坐标系，以及一个具有 19 个状态和 4 个控制

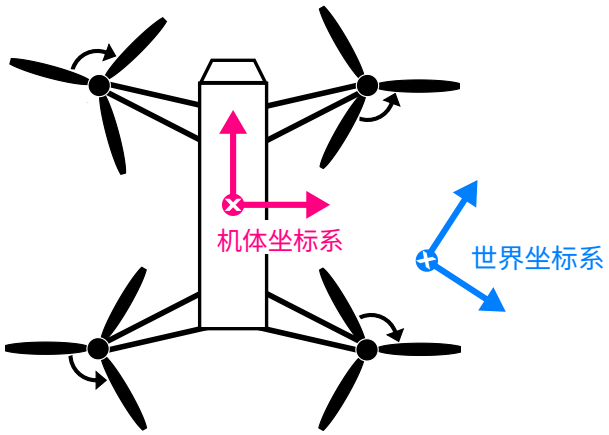

图 2. 坐标系（机体 x 轴指向无人机前方）。

输入：

$$\mathbf{x} = [\mathbf{p}, \mathbf{v}, \lambda, \Omega, \omega, \mathbf{M}_{ext}]^T \quad \mathbf{u} = [u_1, u_2, u_3, u_4]^T$$

状态向量 $\mathbf{x}$ 包含位置 $\mathbf{p} = [x, y, z]$和速度 $\mathbf{v} = [v_x, v_y, v_z]$ ，二者均在世界坐标系中定义。欧拉角 $\lambda = [\phi, \theta, \psi]$ 用于指定机体坐标系的方向，机体坐标系中的角速度为 $\mathbf{\Omega} = [p, q, r]$ ，螺旋桨转速为 $\omega = [\omega_1, \omega_2, \omega_3, \omega_4]$ ，外部力矩扰动为$\mathbf{M}_{ext} = [M_{ext,x}, M_{ext,y}, M_{ext,z}]$ [13]。控制输入$\mathbf{u} = [u_1, u_2, u_3, u_4]$有界 $u_i \in [0, 1]$，其中$u_i = 0$ 和 $u_i = 1$ 分别对应相应螺旋桨的最小（ω_{min}）和最大转速（ω_{max}）。将运动方程（式 1）表示为：

$$f(\mathbf{x}, \mathbf{u}) = \begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{g} + R(\lambda)\mathbf{F} \\ \dot{\lambda} = Q(\lambda)\mathbf{\Omega} \\ I\dot{\mathbf{\Omega}} = -\mathbf{\Omega} \times I\mathbf{\Omega} + \mathbf{M} + \mathbf{M}_{ext} \\ \dot{\omega} = ((\omega_{max} - \omega_{min})\mathbf{u} + \omega_{min} - \omega)/\tau \\ \dot{\mathbf{M}}_{ext} = 0 \end{cases} \tag{1}$$

其中，$I = \text{diag}(I_x, I_y, I_z)$ 为转动惯量矩阵，$\mathbf{g} = [0, 0, g]^T$表示重力加速度，单位为 $g = 9.81\,\text{m s}^{-2}$。旋转矩阵 $R(\lambda)$ 将机体坐标系变换到世界坐标系。我们使用记号 c_θ 和s_ϕ 分别表示对应欧拉角的余弦和正弦。

$$R(\lambda) = \begin{bmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}$$

而 $Q(\lambda)$ 为逆变换矩阵：

$$Q(\lambda) = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix}$$

力 $\mathbf{F} = [F_x, F_y, F_z]^T$ 根据 [17] 中的推力与阻力模型计算。请注意，上标 $\square^B$ 表示机体坐标系，所有模型参数均列于表 I。

$$\begin{aligned} F_x &= -k_x v_x^B \sum_{i=1}^{4} \omega_i \quad F_y = -k_y v_y^B \sum_{i=1}^{4} \omega_i \\ F_z &= -k_\omega \sum_{i=1}^{4} \omega_i^2 - k_z v_z^B \sum_{i=1}^{4} \omega_i - k_h (v_x^{B2} + v_y^{B2}) \end{aligned} \tag{2}$$

以及力矩 $\mathbf{M} = [M_x, M_y, M_z]^T$ 定义为：

$$\begin{aligned} M_x &= k_p(\omega_1^2 - \omega_2^2 - \omega_3^2 + \omega_4^2) + k_{pv} v_y^B \\ M_y &= k_q(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2) + k_{qv} v_x^B \\ M_z &= k_{r1}(-\omega_1 + \omega_2 - \omega_3 + \omega_4) \\ &\quad + k_{r2}(-\dot{\omega}_1 + \dot{\omega}_2 - \dot{\omega}_3 + \dot{\omega}_4) - k_{rr} r \end{aligned} \tag{3}$$

我们采用 [13], 中提出的自适应方法，该方法考虑了力矩方程（公式3）中的模型不匹配，以增强 G&CNET 的鲁棒性。其思路是在学习过程中使用域随机化，假设每条最优轨迹都存在恒定的外部力矩扰动。随后，可以在机载端计算测量力矩与建模力矩之间的差值，并将其输入 G&CNET。

B. 最优控制问题

代价函数 $J(\mathbf{u}\ T)$（公式 4）旨在最小化两个目标：总飞行时间 T和能量 $\int_0^T ||\mathbf{u}(t)||^2 dt$。这两个目标通过混合参数 ϵ进行加权，其中， $\epsilon = 1$ 对应于能量最优飞行，而 $\epsilon = 0$ 对应于时间最优飞行。尽管能量最优项并不直接以最小化时间为目标，但它仍然具有实用价值，因为它能够生成平滑的控制输入，与完全时间最优的“bang-bang”控制曲线相比，为误差留出了更大的容许空间。令 X 表示状态空间，U 表示可行控制集合。所考虑的最优控制问题旨在寻找最优控制策略$\mathbf{u}$ ：$[0,1] \to U$ ，使四旋翼无人机能够从初始条件$\mathbf{x}_0$ 转向一组终端条件 S，同时最小化代价函数 $J(\mathbf{u}\ T)$：

$$\begin{aligned} \underset{\mathbf{u},T}{\text{minimize}} \quad & J(\mathbf{u}, T) = (1-\epsilon)T + \epsilon \int_0^T ||\mathbf{u}(t)||^2 dt \\ \text{subject to} \quad & \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \\ & \mathbf{x}(0) = \mathbf{x}_0 \\ & \mathbf{x}(T) \in S \end{aligned} \tag{4}$$

我们采用与 [12] 类似的流程，使用 Hermite–Simpson 配点法将 OCP 转录为非线性规划（NLP）问题。该 NLP 问题

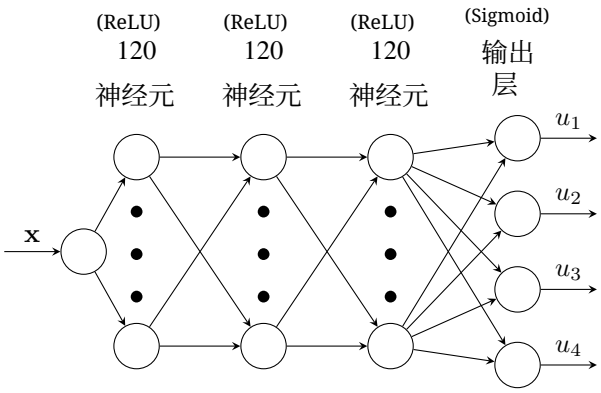

图3. 前馈网络架构。

使用建模语言 AMPL 进行公式化，并离散化为 $N+1$ 个点，再使用名为 SNOPT [20] 的序列二次规划求解器求解。使用 AMPL 是因为它能够向 SNOPT 提供问题的梯度和海森矩阵，从而使求解器更容易收敛。我们将得到的最优轨迹记为$\mathbf{x}_0^* \dots \mathbf{x}_N^*$，相应的最优控制量记为$\mathbf{u}_0^* \dots \mathbf{u}_N^*$。

C. 数据集与学习过程

本研究中的 G&CNET 通过模仿学习/监督学习进行训练，使用由最优状态-动作对构成的数据集，其中状态作为特征，控制量作为标签。因此，数据集中的每个条目为（$\mathbf{x}_i^*, \mathbf{u}_i^*$）$i = 0, \dots, N$。由于最优状态反馈策略的存在性和唯一性，而这源于 Hamilton-Jacobi-Bellman 方程 [14], 因此可以学习最优状态反馈策略 G&CNET($\mathbf{x}_i^*$) $\approx \mathbf{u}_i^*$ 。本研究中始终采用的通用网络架构是前馈神经网络，如图 3 所示。我们使用均方误差作为损失函数：

$$\mathcal{L} = ||\text{G\&CNET}(\mathbf{x}_i^*) - \mathbf{u}_i^*||^2$$

数据集被划分为训练数据（80%）和验证数据（20%）。网络的权重使用 Adam优化器 [21] 进行更新，且不采用权重衰减。初始学习率设置为 $l = 0.1 \cdot 10^{-2}$，并使用调度器：每当验证数据集上的损失经过 $p = 6$ 个训练轮次后趋于平稳，就将学习率按 $f = 0.9$ 的因子降低。为了促进学习过程，对特征进行归一化处理，并在末尾添加一个额外的输出层，将控制输入 $u_i \in [0,1]$ 映射到相应的转/分钟。

D. 实验设置

本研究中的实验平台是 Parrot Bebop 1 四旋翼无人机与开源 Paparazzi UAC软件 [22] 的组合。这款无人机专为自拍无人机市场设计，因此并非竞速无人机。由于其最大推重比为 ~ 1.7 ，它无法达到自主无人机竞速相关文献中无人机所能达到的相同速度。然而，这也意味着它将允许

k_x	k_y	k_ω	k_z	k_h	I_x	I_y	I_z
1.08e-05	9.65e-06	4.36e-08	2.79e-05	6.26e-02	0.000906	0.001242	0.002054
k_p	k_{pv}	k_q	k_{qv}	k_{r1}	k_{r2}	k_{rr}	τ
1.41e-09	-7.97e-03	1.22e-09	1.29e-02	2.57e-06	4.11e-07	8.13e-04	0.03

表 I Parrot Bebop 1 四旋翼无人机的模型参数。转动惯量取自 [18], , 所有其他参数取自 [13]。

使我们能够在可用的相对狭小飞行空间内让其控制量达到饱和，而且用它进行测试也更安全。这些特性正是本研究选择 Parrot Bebop 1 的主要原因。Bebop 配备了 MPU6050惯性测量单元和 Parrot P7 双核 Cortex A9中央处理器，我们用它在无人机上实时运行代码。此外，四旋翼无人机还可以测量四个螺旋桨的角速度。我们在代尔夫特理工大学航空航天工程学院的 The Cyberzoo 内进行飞行测试。The Cyberzoo 是一间 10 米 × 10 米的实验室，配备了动作捕捉系统（Optitrack）。我们实时向无人机发送位置、速度和姿态测量值，并使用扩展卡尔曼滤波器将这些数据与惯性测量单元的测量值融合，为 G&CNET 提供准确的状态估计。随后，G&CNET 直接输出转/分钟指令并将其发送至电机。对于微分平坦性控制器，我们使用 INDI 控制器跟踪参考轨迹。

III. 时间最优四旋翼无人机飞行

在本节中，我们训练 G&CNET，以近似不同的控制策略，从完全能量最优（$\epsilon = 1.0$）到完全时间最优（$\epsilon = 0.0$）。我们通过 $\ddot{\mathbf{x}} = f(\mathbf{x}, \text{G\&CNET}(\mathbf{x}))$ 仿真 G&CNET 的响应，并表明，对于接近时间最优的飞行，达到较低的损失 $\mathcal{L}$ 不仅变得更加困难，而且也更加关键，因为与能量最优飞行相比，四旋翼无人机纠正误差的控制权更小。

数据集和网络架构

我们通过在以下范围内对每条轨迹的初始条件进行均匀采样来生成所有训练数据集（其中目标航路点为参考点，参见附录VII-C中图19的顶部图示作为示例）：

$$
\begin{aligned}
x_0 &\in [-5.0, -2.0], \mathrm{m} & y_0 &\in [-1.0, 1.0], \mathrm{m} \\
z_0 &\in [-0.5, 0.5], \mathrm{m} & v_{x_0} &\in [-0.5, 5.0], \mathrm{m\,s^{-1}} \\
v_{y_0} &\in [-3.0, 3.0], \mathrm{m\,s^{-1}} & v_{z_0} &\in [-1.0, 1.0], \mathrm{m\,s^{-1}} \\
\phi_0 &\in [-40, 40], \mathrm{deg} & \theta_0 &\in [-40, 40], \mathrm{deg} \\
\psi_0 &\in [-60, 60], \mathrm{deg} & p_0 &\in [-1, 1], \mathrm{rad\,s^{-1}} \\
q_0 &\in [-1, 1], \mathrm{rad\,s^{-1}} & r_0 &\in [-1, 1], \mathrm{rad\,s^{-1}}
\end{aligned}
$$

转速限制分别设置为 $\omega_{min} = 3000$ 和 $\omega_{max} = 12000$，初始转速的采样范围为：

$$\omega_{i_0} \in [-\omega_{min}, \omega_{max}], \mathrm{RPM} \quad i = 1, \ldots, 4$$

对于每条轨迹，我们假设存在恒定的外部力矩扰动，其采样范围为：

$$
\begin{aligned}
M_{ext,x} &\in [-0.04, 0.04], \mathrm{N\,m} \\
M_{ext,y} &\in [-0.04, 0.04], \mathrm{N\,m} \\
M_{ext,z} &\in [-0.01, 0.01], \mathrm{N\,m}
\end{aligned}
$$

我们将每条轨迹的期望终端状态设为：$\mathbf{p}_f = \mathbf{0}\,\mathrm{m}$、$\psi_f = 45°$、$\boldsymbol{\Omega}_f = \mathbf{0}\,\mathrm{rad\,s^{-1}}$ 以及 $\dot{\boldsymbol{\Omega}}_f = \mathbf{0}\,\mathrm{rad\,s^{-2}}$。终端速度$\mathbf{v}f$ 受到约束，其方向与终端航向 $\psi_f = 45°$一致，而终端速度大小及其余所有状态则不受约束。之所以选择这些特定约束，是因为在四旋翼无人机抵达目标航路点之前向右移动该航路点，便可使 G&CNET 飞行多种仅包含右转的航迹。第V节将更详细地介绍单航路点与连续航路点飞行。每个数据集包含 10,000 条最优轨迹，均在 $N = 199$个点进行采样。我们选择对每个独立的 G&CNET 训练$p = 10$ 轮次，训练批次大小为 256。图 4 展示了对于不同的 ϵ，四个旋翼之一的典型最优控制输入 u^* 如何变化。在相同的初始和最终条件下，u^* 会从平滑的控制曲线（$\epsilon = 1.0$）变化为所谓的“bang-bang”控制曲线（$\epsilon = 0.0$），以实现完全的时间最优飞行；在这种情况下，四个旋翼中至少有一个始终处于饱和状态。请注意，时间最优解完成这条轨迹仅需 1.2s，而能量最优解则需 1.65s。

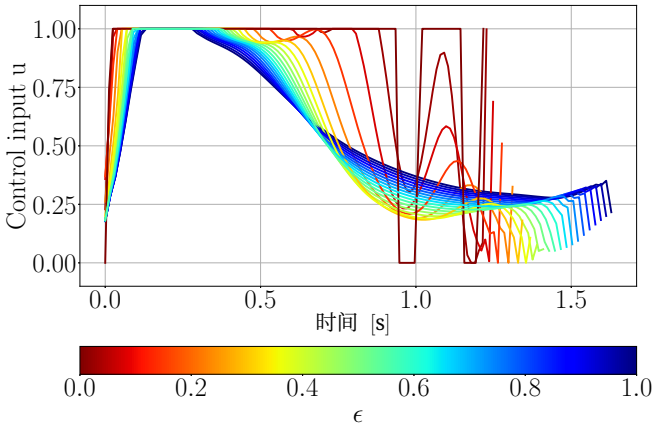

图 4. 不同 ϵ 下的最优控制输入 u^*（仅显示一个旋翼）。所有轨迹具有相同的初始和最终条件。

结果与讨论

我们在表 II 中报告了所得的损失 $\mathcal{L}$ 以及相应的平均控制误差 [%]。显然，随着我们通过减小 $\mathcal{L}$ 来提高时间-

最优目标函数（式 4）的权重，损失 $\mathcal{L}$ 会上升。时间最优策略之所以更难学习，是因为其切换时间数量较多，如图 4 所示。所得拓扑结构的较大梯度相比能量最优飞行所采用的平滑连续控制曲线，更难以精确近似。

在实践中，可以通过扩大训练数据集、延长训练时间，最终增大网络规模来缓解这一问题。不过必须指出，更大的网络会直接影响在无人机机载运行网络推理的频率。此外，对于时间最优飞行，仅降低损失 $\mathcal{L}$ 并不能解决所有问题。时间最优飞行意味着无人机在其飞行包线的边缘运行，几乎没有控制余量。这意味着，与能量最优飞行相比，从微小的控制误差中恢复会困难得多。

我们使用 Scipy（ 5 [23]阶显式 Runge-Kutta 积分方法）对 $\dot{\mathbf{x}} = f(\mathbf{x}, \text{G\&CNET}(\mathbf{x}))$ 进行数值积分，以仿真测试所有 G&CNET。仿真中未添加任何扰动，并采用与求解最优轨迹时相同的运动方程和执行器延迟。求解器会自行选择步长，且未实现零阶保持。即便如此，要维持$\epsilon = 0.1$ 和 $\epsilon = 0.0$的稳定飞行仍然很困难。这表明，对于接近时间最优的飞行，$\mathcal{L} = 1.12 \cdot 10^{-3}$ 的训练损失（平均控制误差为 $\pm 3.35\%$）是维持稳定性的上限。

ϵ	损失 $\mathcal{L}$	控制误差 [%]
1.0	$1.24 \cdot 10^{-4}$	± 1.12
0.95	$1.41 \cdot 10^{-4}$	± 1.19
0.9	$1.38 \cdot 10^{-4}$	± 1.18
0.85	$1.35 \cdot 10^{-4}$	± 1.16
0.8	$1.46 \cdot 10^{-4}$	± 1.21
0.75	$1.48 \cdot 10^{-4}$	± 1.22
0.7	$1.60 \cdot 10^{-4}$	± 1.27
0.65	$1.72 \cdot 10^{-4}$	± 1.31
0.6	$1.97 \cdot 10^{-4}$	± 1.40
0.55	$1.79 \cdot 10^{-4}$	± 1.34
0.5	$2.05 \cdot 10^{-4}$	± 1.43
0.45	$2.37 \cdot 10^{-4}$	± 1.54
0.4	$2.99 \cdot 10^{-4}$	± 1.73
0.35	$2.96 \cdot 10^{-4}$	± 1.72
0.3	$3.39 \cdot 10^{-4}$	± 1.84
0.25	$4.82 \cdot 10^{-4}$	± 2.20
0.2	$5.25 \cdot 10^{-4}$	± 2.29
0.15	$7.08 \cdot 10^{-4}$	± 2.66
0.1	$1.12 \cdot 10^{-3}$	± 3.35
0	$7.01 \cdot 10^{-3}$	± 8.37

表 II

验证数据集上的均方误差及
针对训练于.....的 G&CNET 的对应控制误差
不同的代价函数。

IV. 考虑螺旋桨变化的最大角速度

本节中，我们通过考察低于和超过可达最大转速限制 ω_{max} 对鲁棒性的影响，探究以可达最大转速限制飞行的重要性

飞行过程中。旋翼的最大角速度一开始可能就被错误识别，也可能在飞行期间因螺旋桨承受的气动载荷变化而暂时改变，甚至会随着电池电量耗尽而逐渐下降。我们在一次持续 6 分钟的 Parrot Bebop 1 测试飞行中，观察到 $\dot{\omega}_{max} = -1$ 转/分钟$^{-1}$ 稳步下降（见附录 VII-B）。时间最优飞行的特点是，控制曲线会在相当长一段时间内使旋翼达到饱和状态。请看图5，其中展示了在一次使用 $\epsilon = 0.35$ 的真实飞行中，单个旋翼的指令角速度与观测角速度。无法达到转速上限 $\omega_{max} = 12000$。这引出了一个问题：以正确的 ω_{max}飞行有多重要?

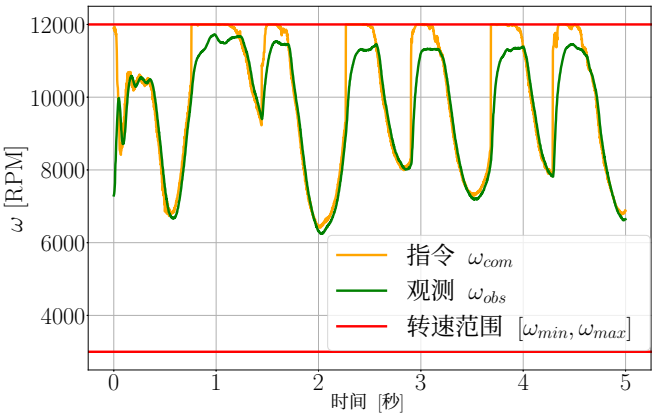

图5. 使用 $\epsilon = 0.35$ 进行真实飞行期间，指令角速度与观测角速度 ω 之间的差异（仅显示一个旋翼）。

我们来考虑一个 $\omega_{max}[10000, 12000]$ 转/分钟的范围。图 6 显示了在时间最优（$\epsilon = 0.0$）着陆情况下，不同 ω_{max} 对应的最优控制解有何差异。四旋翼无人机从 5m 高度的悬停状态出发，需要达到以下终端条件：$\mathbf{p}_f = \mathbf{0}$、$\mathbf{v}_f = \mathbf{0}$、姿态 $\lambda_f = \mathbf{0}$ 以及角速度 $\mathbf{\Omega}_f = \mathbf{0}$。在此情况下，不施加外部力矩扰动$\mathbf{M}_{ext}$ 。鉴于该最优控制问题具有对称性，四个旋翼的最优控制解相同。当 $\omega_{max} = 12000$时，旋翼开始饱和的时间比 $\omega_{max} = 10000$时晚 0.1s。此处的最优控制解从某个切换点开始直到结束，均施加最大制动力，因此高估 ω_{max} 总会导致坠毁，因为四旋翼无人机开始制动的时间会太晚。这个情况还表明，仅将螺旋桨的观测角速度 ω 作为状态反馈，并不足以减轻错误识别 ω_{max} 的影响，因为当无人机观察到旋翼未达到所需的 ω_{max}时，为时已晚。最后，要维持稳定飞行，必须精确设定最小与最大控制输入之间的切换时间，

尤其是因为随着 ϵ趋近于零，几乎不再剩余任何控制权。

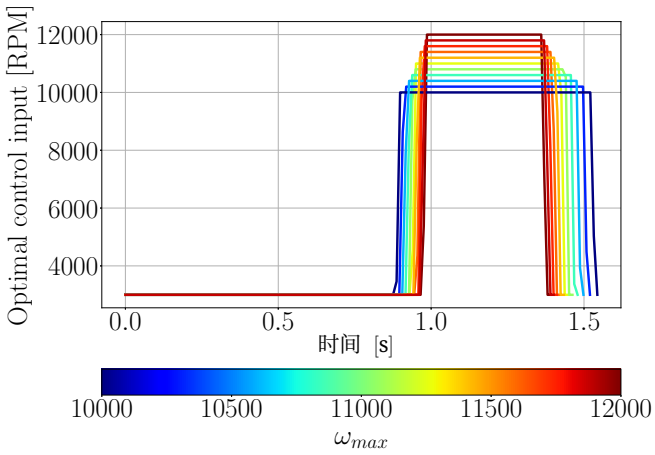

图 6. 不同 ω_{max} 下的最优控制输入 [转/分钟] （仅显示一个旋翼）。所有轨迹都具有相同的初始和最终条件。目标是从 5 米高处进行时间最优（$\epsilon = 0.0$）着陆。鉴于该最优控制问题具有对称性，所有旋翼接收相同的控制输入。

峰值跟踪算法

当 ω_{max} 的初始猜测设置得过高时，一旦新上限变得可观测，就可以通过机载系统立即识别该上限。我们提出了一种峰值跟踪算法：每当积分 $\int_{t-\Delta t}^{t}(\omega_{exp} - \omega_{obs})\, dt$ 超过某个阈值 p_{thresh}时，就将ω_{max} 设置为观测到的最高 ω_{obs}，参见图 7 中的 $t = 0.21$s。预期的 ω_{exp} 可以通过对指令 ω_{com} （公式 1）施加一阶延迟，在无人机机载端进行计算。根据转速范围$[\omega_{min}, \omega_{max}]$，需要调整的算法参数包括：计算积分并记录观测到的 ω_{obs} 最后一个峰值所采用的时间窗口 Δt，以及触发 ω_{max}变化的阈值 p_{thresh}。值得注意的是，另一种显而易见的方法是将 ω_{max} 建模为四旋翼无人机状态$\mathbf{x}$ 及其电池电压的函数。然而，这种方法要求每次更换无人机时都进行系统辨识。我们算法的优势在于，只要能够通过机载系统测量螺旋桨的角速度，就可以轻松应用于任何无人机。此外，我们的算法还能够考虑因任何无法建模的意外故障而导致的较低ω_{max}。当前的设置只能在发生超调后才能修正 ω_{max}，否则该上限无法被观测到。我们在仿真环境和真实四旋翼无人机上均测试了 RPM 峰值跟踪算法。在这两种情况下，当初始 ω_{max} 过高时，只需在某个旋翼达到饱和后不到 $\Delta t = 0.13$s，就能修正该上限。我们还通过仿真测试了峰值跟踪算法能否持续修正缓慢下降的 ω_{max} （例如，由于电池电压下降所导致的变化）。峰值跟踪算法始终能够将识别出的上限与正确的 ω_{max} 之间的误差保持在 70 转/分钟以内。

自适应 *G&CNET*

我们考虑在一系列具有不同 ω_{max}的最优轨迹上训练 G&CNET，而不是仅仅学习.....

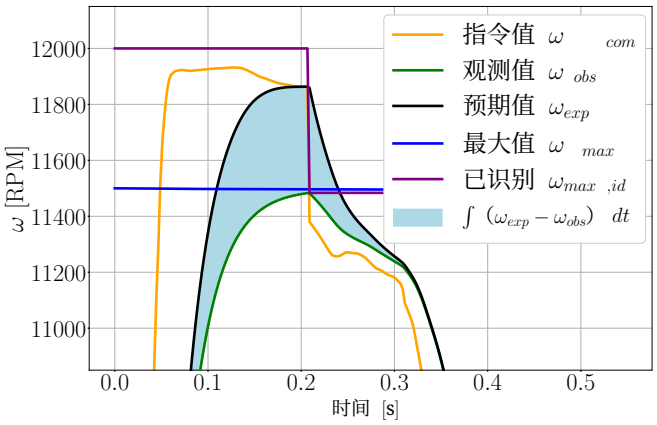

图 7. 使用峰值航迹算法识别当前 ω_{max} （蓝线）。

状态 $\mathbf{x}$ 与控制量 $\mathbf{u}$之间的映射，我们将 ω_{max} 作为一个特征。这使 G&CNET 能够在飞行过程中利用峰值跟踪算法的输出作为额外输入进行自适应，新的架构见图 8。

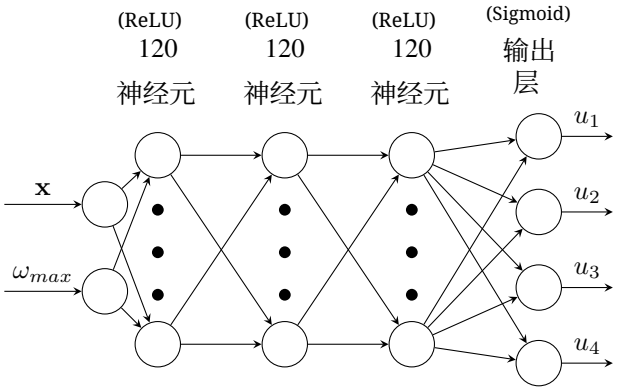

图 8. 针对 ω_{max} 变化的自适应 G&CNET 前馈网络架构。

结果与讨论

我们首先在仿真中评估超调和欠调的影响ω_{max}。我们使用与第 III 节相同的初始和最终条件边界训练两个自适应 G&CNET，但每个训练数据集现在包含 100,000 条最优轨迹，并且 $\epsilon = 0.4$。第一个 G&CNET 使用这样的最优轨迹进行训练：其中 ω_{max} 在 $[10000, 11000]$ 转/分钟范围内均匀采样；第二个 G&CNET$[11000, 12000]$ 转/分钟。请注意，对于这项分析，也可以训练多个 G&CNET，每个网络仅针对 ω_{max}的一个取值进行专门训练。自适应 G&CNET 使我们能够更快地直观了解改变 ω_{max} 会如何影响其性能。我们另行生成了一个评估数据集，其中包含 10,000 条最优轨迹，且所有轨迹均使用$\omega_{max} = 11100$ 转/分钟。选择这个 ω_{max} 值是为了在接近 Parrot Bebop 1 实际限制的同时，模拟超出和低于该限制的较大范围。鉴于这些最优轨迹并非真正的解析最优解，而是通过直接法求解得到的，因此可以预期会因节点间的

积分误差而产生相当大的数值噪声。为减轻部分噪声，我们采用节点加倍技术扩充全部 10,000 条最优轨迹。如果没有良好的初始猜测，这里考虑的最优控制问题过于复杂，SNOPT 无法在 $N > 400$ 个节点的情况下收敛。因此，我们先对 $N = 100$求解最优控制问题，再使用二次样条插值解$\mathbf{x}^*$ 和$\mathbf{u}^*$，最后将插值结果投影到新的节点网格（例如 $N = 200$）上，为求解器提供良好的初始猜测。重复这一过程后，我们生成了包含 10,000 条轨迹的高保真评估数据集，每条轨迹有$N = 1000$ 个节点（对于长度为 5 米的最优轨迹，这相当于相邻节点间距为 5 毫米）。图 9 展示了从这 10,000 条最优轨迹得出的平均位置误差 [厘米]；这些误差是在以评估数据集中轨迹的相同初始条件为起点，仿真 G&CNET 的响应$\mathbf{x} = f(\mathbf{x}, \text{G\&CNET}(\mathbf{x}))$时计算的。通过手动更改网络的输入 ω_{max}，我们可以仿真其对偏离最优轨迹程度的影响。即使 G&CNET 知道正确的限制值（图 9 中间的箱线图），其平均位置误差仍约为 4 厘米，这是训练期间损失不为零所致。超调和欠调 ω_{max}也呈现类似趋势：差异越大，平均位置误差越大。这表明，对 ω_{max} 的错误识别会影响飞行的鲁棒性。

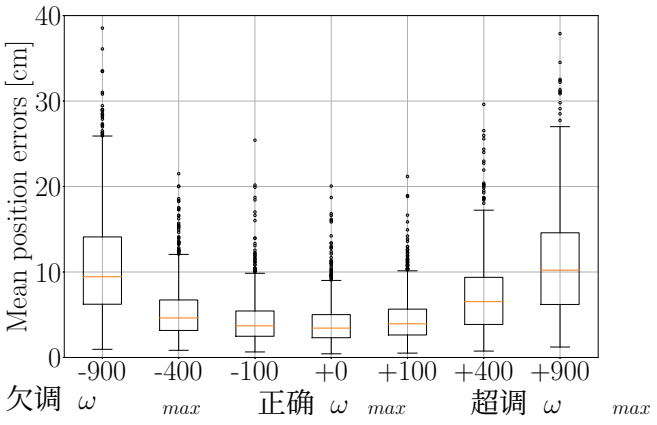

图 9. 10,000 条最优轨迹（每条轨迹有 1000 个节点）的平均位置误差为 [cm]。对于每个箱线图，G&CNET 的结果要么低于、要么高于、要么恰好等于正确的 ω_{max} [转/分钟]。

然而，与我们的预期相反，即使 ω_{max} 存在较大偏差，也不是造成现实差距的最关键因素。在仿真和真实四旋翼无人机上的实验中，即使 ω_{max} 偏差达到 $+1000$ 转/分钟，我们仍能成功飞行。不过，随着我们逐渐接近时间最优飞行，容错空间会越来越小，因此尽可能贴近最优轨迹飞行对于增强鲁棒性至关重要。

请看图 10，其中展示了使用相同 G&CNET（$\epsilon = 0.5$）进行的三次真实飞行。该 G&CNET 使用 60,000 条最优轨迹进行训练，且 ω_{max} 在 $[10500, 13000]$ 转/分钟范围内均匀采样。我们人为限制螺旋桨的最大角速度为 $\omega_{max} = 11000$转/分钟。左侧轨迹展示了 G&CNET 将正确的 ω_{max} 作为输入时产生的飞行结果，另外两条轨迹对应的飞行中，G&CNET 接收了错误的 ω_{max} 作为输入（超调量分别为 $+500$ 转/分钟和 $+1000$转/分钟）。超调指的是 G&CNET 假定 ω_{max} 高于其实际值。这三种情况下的差异主要体现在第一圈。G&CNET 从悬停状态迅猛起步并向下俯仰 $\theta = -85°$，其第一个转弯与最优路径有显著偏差；超调ω_{max}越大，偏差越明显。

一种提高飞行鲁棒性的方法是将同一个 G&CNET 与峰值跟踪算法结合使用。我们将最大转/分钟的初始猜测设为 $\omega_{max} = 12000$，并且不人为限制螺旋桨的最大角速度。对于 Bebop 1，螺旋桨的实际物理极限为 $\omega_{max} = 11300$，因此我们超出该极限 $+700$ 转/分钟。飞行开始后 0.2 秒，以及首个旋翼饱和后 0.1 秒，峰值跟踪算法便能识别出正确的极限（$\omega_{max} = 11300$），并将其传递给 G&CNET。由此得到的飞行结果如图 11 所示。尽管起初超出正确极限 $+700$转/分钟，四旋翼无人机偏离最优路径的程度仍小于图 10 所示的情况。

这一领域还需要更多研究，可以设想，对于更激进的飞行（例如 $\epsilon = 0.0$），高估ω_{max} 可能会变得更加关键。回想图 6，如果旋翼饱和晚了 0.1 秒，那么以 10 米/秒$^{-1}$飞行的四旋翼无人机将在最优制动点后 1 米处才开始制动。如前所述，我们发现，选择用于训练 G&CNET 的 ω_{max} 取值范围会影响网络的整体飞行表现。例如，使用在 $[10500, 13000]$转/分钟范围内均匀采样的 ω_{max} 进行训练的 G&CNET，其飞行圈次的稳定性不如使用较小 ω_{max} 范围（例如 $[11000, 12000]$ 转/分钟）训练的网络。这可能表明，需要重新审视当前的网络架构，以便更准确地学习控制策略。从图 10、11 可以看出，G&CNET 在最初几圈中难以飞过第一个航点。

V. 连续航点飞行

[9] 的作者分析了人类无人机飞行员的视线，发现他们会提前观察多个门，而不是只盯着下一个门。此前的工作 [12], [13]主要研究如何在四旋翼无人机上训练并部署 G&CNET，使其从空间中的一个点飞往指定航点。通过巧妙设置最优控制问题的终端条件，可以在四旋翼无人机即将抵达航点前切换航点的位置，使其连续飞行。显而易见的下一步是使用两个连续航路点训练 G&CNET。全局最优轨迹会针对完整航迹进行优化，同时考虑所有航路点。然而，相比仅针对单个航点的方法，在至少包含两个航路点的视界上进行优化，能使无人机的飞行更接近最优。我们提出了一种实现连续航点飞行的方法，并展示了这种制导策略的一些优势。

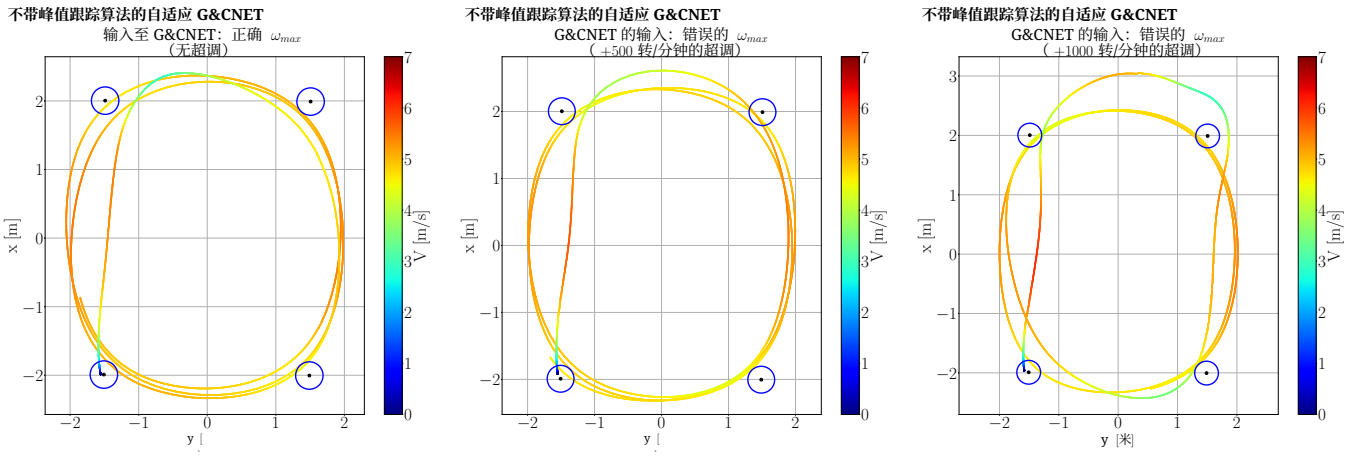

图 10. 搭载 $\epsilon = 0.5$ 的真实飞行轨迹（俯视图）。左：无 ω_{max} 超调。中： ω_{max} 超调 +500 转/分钟。右： ω_{max} 超调 +1000 转/分钟。训练期间，最优轨迹需要经过一个半径为 20cm 的球体范围内，该范围由蓝色圆圈标示。详情请参见第V节。

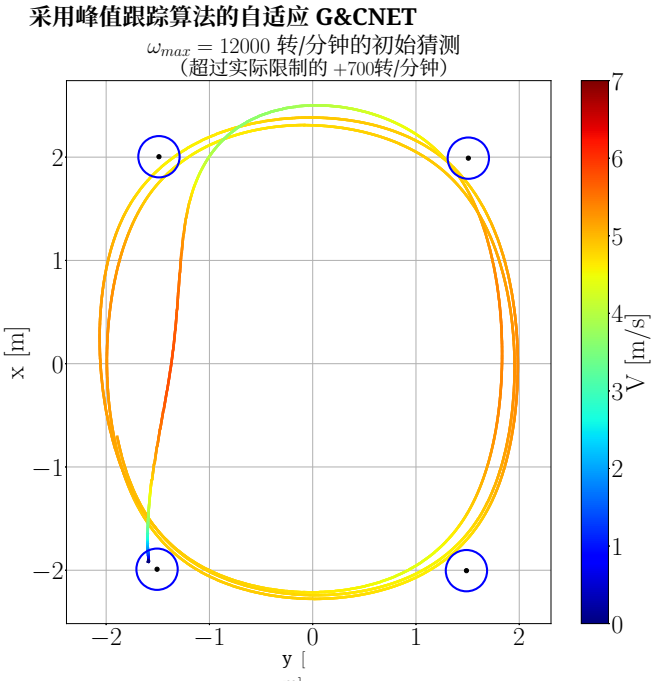

图 11. 使用峰值跟踪算法和自适应 G&CNET 的 $\epsilon = 0.5$ 真实飞行轨迹（俯视图）。 ω_{max} 的初始猜测偏差为 +700 转/分钟。训练期间，最优轨迹需要经过半径为 20cm 的球体范围内，该范围以蓝色圆圈表示。详情请参阅第V节。

学习复杂赛道的方法

考虑以时间最优的方式穿越一组门这一任务。通过一道门的最优方法，很大程度上取决于下一道门的位置和朝向。为了根据两个连续门的相对位置和朝向学习不同的最优轨迹，我们在 AMPL 中添加了一项中间约束。该约束要求最优解经过一个以$\mathbf{WP}_{1,pos} = [\mathrm{WP}_{1,x}, \mathrm{WP}_{1,y}, \mathrm{WP}_{1,z}]$为中心的球体。虽然本研究没有使用无人机的摄像头，但无人机的航向 ψ 始终指向一个方向、确保下一个航点位于摄像头的视场内，这一点十分重要。因此，我们还添加了一项用于约束航向的项 $\mathrm{WP}_{1,psi}$ at 中间航点：

$$\mathrm{WP}_{1,threshold} \geq (x_i^* - \mathrm{WP}_{1,x})^2 + (y_i^* - \mathrm{WP}_{1,y})^2 + (z_i^* - \mathrm{WP}_{1,z})^2 + (\psi_i^* - \mathrm{WP}_{1,psi})^2$$

我们选择这一约束，是因为它允许放宽中间航点处的位置误差和航向误差。对于复杂航迹，很难提前确定任意阶段的最优路径和航向。以这种方式实施约束，可以使求解器根据初始条件以及两个航点的相对位置，拥有一定的自由度，从而以更优的方式通过中间航点。图 19（附录VII-C）中的下方图示展示了采用中间航点约束时的最优轨迹。由于此时的轨迹平均长度比单航点飞行时更长，因此我们以 $N = 319$ 个点对其进行采样。我们通过将两个即将到来的航点的相对位置作为网络架构的输入，向G&CNET提供相关信息，参见图12。$\mathbf{WP}_{rel}$ 由于网络架构所能承载的信息量已接近上限，因此我们不在此处将 ω_{max} 作为额外输入添加到网络架构中。为了始终达到预期的饱和度，我们采用保守的飞行方式，以四旋翼无人机能够达到的 ω_{max} 进行飞行。

这种方法非常适合学习如何飞行更复杂的赛道，因为可以生成不同类型的最优转弯。附录 VII-A 中，我们提供了在仿真中飞行八字形赛道的轨迹。训练数据集既可以针对已知赛道量身定制，也可以像我们将在本节末尾看到的那样，覆盖两个航点之间的一系列相对位置，使 G&CNET 能够飞行不同形式的赛道。

与能量最优单航点飞行的比较

学习在考虑多个航点的情况下进行飞行的另一个优势是，G&CNET 能够在整个航迹上实现更优的飞行。我们通过在 Bebop 上进行两次真实飞行来展示这一点，并采用能量最优

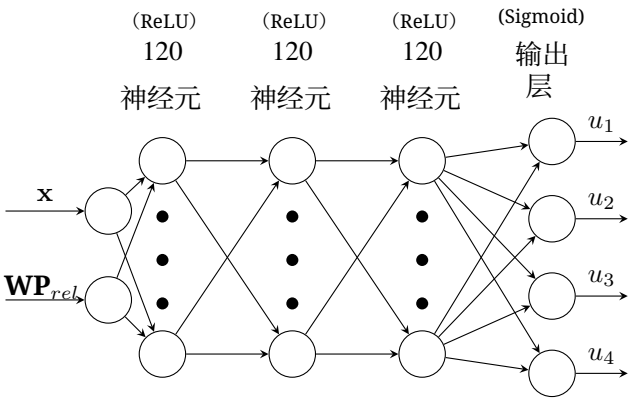

图12。使用多个航路点的 G&CNET 前馈网络架构。我们添加了输入 $\mathbf{WP}_{rel}$，用于告知网络即将到达的两个航路点的相对位置。

G&CNET，随后计算随时间变化的代价（含$\epsilon = 1.0$的式(4)）。这里选择能量最优控制问题，是因为这是 G&CNET 最容易学习的最优控制策略。对于不同的 ϵ，也可以进行相同的分析。图13展示了所得轨迹。左图显示的是仅针对单个航点训练的 G&CNET。当四旋翼无人机与航点之间的三维欧几里得距离小于 1.2m[13]时，我们切换至下一个航点。我们尝试了不同的切换距离。总体而言，切换距离越大，四旋翼无人机切过弯道的程度越大。对于较小的切换距离，航点处的位置误差会变小；然而，由于 G&CNET 会尝试完全满足最优控制问题的终端条件，四旋翼无人机会减速。请注意，我们采用自由的终端速度大小，仅约束速度向量的方向，使其与期望终端航向角对齐（$\psi_f = 45°$）。另一个 G&CNET（中图）是在两个连续航路点 $\text{WP}_{1,threshold} = 0.2$ 和 $\text{WP}_{1,psi} = 45°$上训练的。这意味着，最优轨迹必须以大约 $\psi = 45°$的航向通过半径为 20cm 的球体。该 G&CNET 使用包含两“类”转弯轨迹的数据集进行训练：两个航点相距 3m 的轨迹，以及相距 4m 的轨迹。因此，只需向 G&CNET 额外输入一个参数（图12中的$\mathbf{WP}_{rel}$），即可告知网络即将出现两种可能转弯中的哪一种。针对单航点和连续航路点情况下用于训练 G&CNET 的最优轨迹可视化结果见附录VII-C。与单航点飞行相比，我们不再需要在航点处的位置误差与速度之间进行权衡。每当 G&CNET 通过一个航点时，我们就切换航点，因此在第一次转弯时，是在最终航点前 3m 处切换；下一次转弯则是在最终航点前 4m 处切换，依此类推。我们计算这些飞行过程中的代价 $\int_0^T ||\mathbf{u}(t)||^2 dt$，并将其绘制在最右侧的图中（图13）。在两个连续航路点上训练的网络消耗的能量更少，因此随时间变化的代价函数得到了更好的最小化。我们还注意到，控制输入更加平滑并且在连续航点飞行的情况下，饱和程度较低，从而为纠正误差保留更多控制权。这可能是因为，在单个航点上训练的 G&CNET 会不必要地使旋翼饱和，因为它会进行更多的减速和加速。

与最小化 *snap* 基准的比较

我们还考虑使用 G&CNET 尽可能快地飞完 4×3 m 航迹。我们选择了广为人知的基于微分平坦性的控制器（DFBC） [11] 作为基准，以便与我们的 G&CNET 进行比较。这种最先进的控制器通过最小化 snap（位置的四阶导数），利用多项式生成平滑轨迹。随后，外环增量非线性动态逆（INDI）控制器对参考轨迹进行跟踪。与能量最优情况一样，G&CNET 在两种轨迹上进行训练：一种是两个航点相距 3m，另一种是两个航点相距 4m，只不过现在 $\epsilon = 0.5$。最终的飞行结果如图14所示。由于 G&CNET 始终尝试穿过图14中蓝色圆圈的顶点，因此我们在 DFBC 的多项式生成过程中将航点向内移动，使其与这些顶点重合，从而使比较更加公平。两种控制器各有优势，例如，G&CNET 能够执行更加激进的机动。这一点在第一圈中尤为明显，G&CNET 用时 3.22s（DFBC 用时 3.46s）。然而，DFBC 在起飞阶段的瞬态行为结束后，能够维持更高的速度。DFBC 完成第二圈用时 2.7s，而 G&CNET 用时为 2.88s。这些飞行过程中的指令角速度和观测角速度见附录 VII-D。G&CNET 的控制输入明显比 DFBC 的更加平滑。需要注意的是，这两种控制器仍有改进空间。可以在 DFBC 所使用的 INDI 控制分配中引入加权最小二乘法。此外，在此次比较中，G&CNET 具有不公平的优势，因为它是在 $\omega_{max} = 11300$ RPM 的条件下训练的，这一数值接近 Bebop 的真实极限。然而，DFBC 假设其转速为 $\omega_{max} = 12000$RPM。最后，通过 G&CNET 还可以实现更快的飞行（更低的 ϵ），但代价是更大的位置误差和更不稳定的飞行。

G&CNET 的灵活性

最后，我们强调 G&CNET 相比 DFBC 的一项主要优势：能够在线重新计算轨迹及相应的最优控制量。G&CNET 非常灵活，即使偏离全局最优路径，也能立即计算出新的最优控制量 $\mathbf{u}^*$（只要训练数据集对四旋翼状态的表示足够精确）。这也意味着 G&CNET 能够处理训练数据中的不同航点位置。另一方面，DFBC 只能依赖离线生成的轨迹。DFBC 总会尽可能贴近这条轨迹。但这会带来问题，因为受现实情况影响，偏离这条轨迹在所难免。

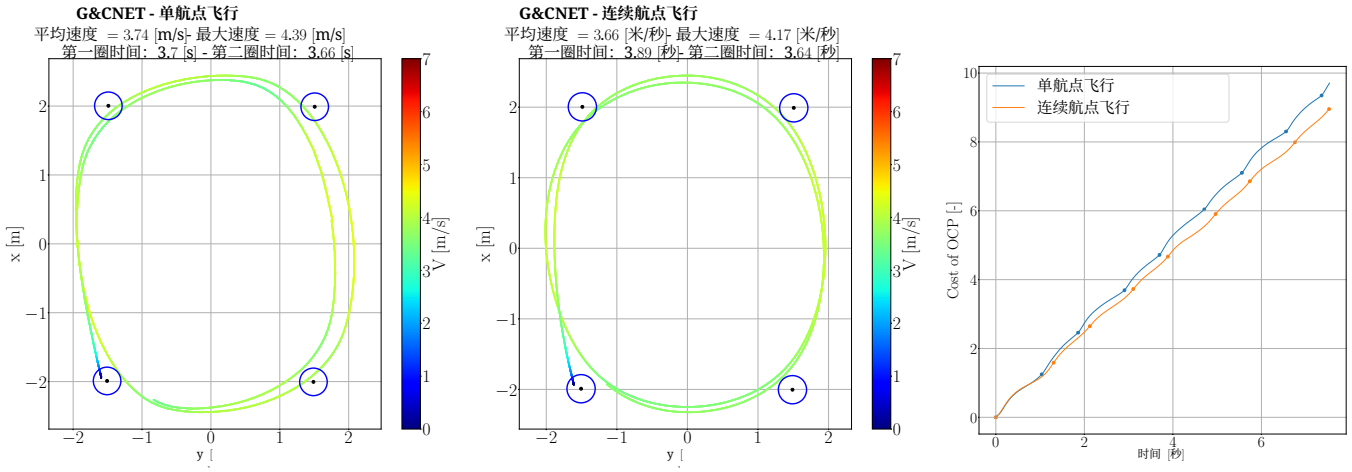

图 13. 使用能量最优 G&CNET 的真实飞行轨迹（俯视图）。左：单航点飞行。中：连续航点飞行。右：两次飞行的代价函数随时间的变化。圆点表示切换至下一个航点（或多个航点）的切换时间。蓝色圆圈表示中间航点处的位置约束。

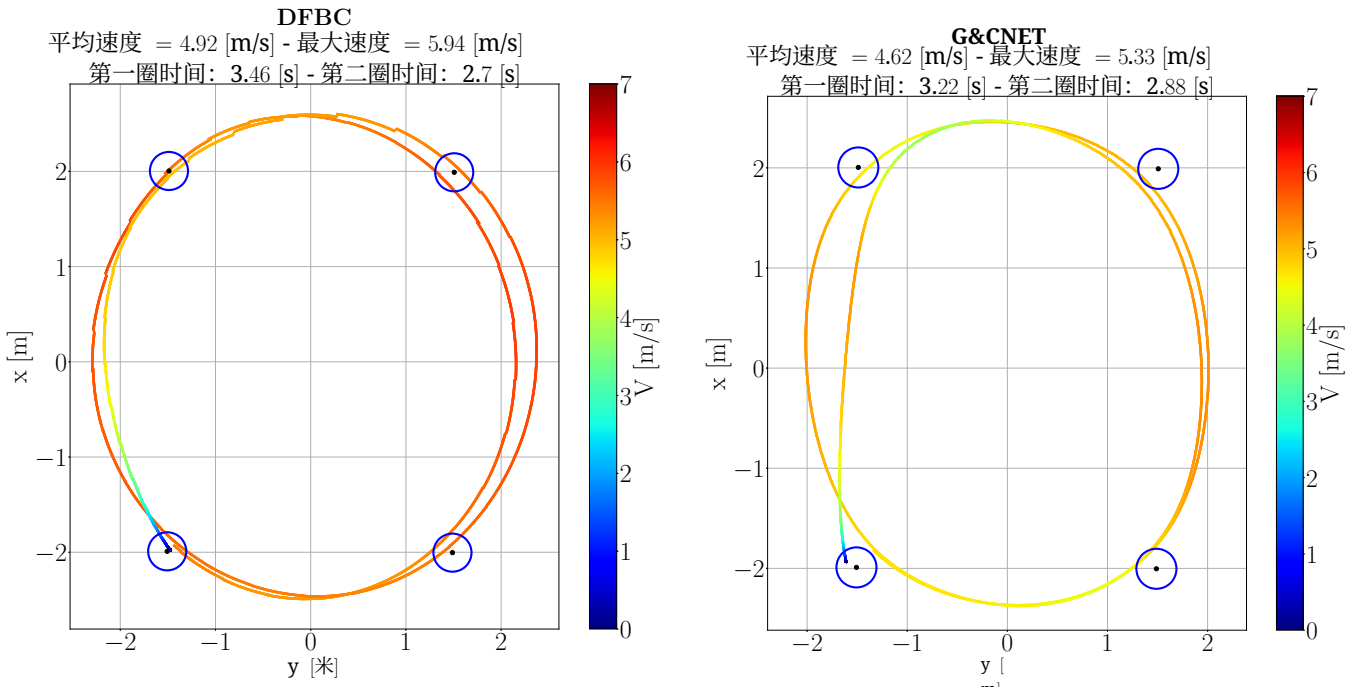

图 14. 左：轨迹（俯视图）采用 DFBC（最小快变导数）的真实飞行轨迹（俯视图）。右：真实飞行轨迹（俯视图） $\epsilon = 0.5$。G&CNET 正在训练 ed 于 2 个连续航点上。蓝色圆圈表示 G&CNET 在中间航点处的位置约束。

间隙。一旦偏离，这条轨迹便不再是最优轨迹，因此应重新计算一条新的轨迹。移动航点也需要离线计算新的最优轨迹。最后，由于网络规模相对较小，采用当前网络架构时，G&CNET 可在 Bebop 机载设备上以 450 Hz 的频率进行推理。这两项特性都使 G&CNET 能够抵消近似误差，而不是让这些误差随时间累积。

我们研究飞行 4×3m 航迹的任务，但现在四个航点会在 XY 平面内的 1m^2 正方形范围内随机移动（在 G&CNET 将其纳入考量之前；见图 15 中的虚线正方形）。我们通过在 1m^2 的正方形范围内均匀采样 $\text{WP}_{1,x}$, $\text{WP}_{1,y}$ ，针对一系列相对航点位置训练网络。

以最终航点为中心，距离为3.5米。本实验中，为简化处理，保持高度恒定。由于相对航点位置现在可以在两个维度上变化，因此需要增加两个额外输入（图12中的$\mathbf{WP}_{rel}$ ）以向网络提供相关信息。我们训练了两个G&CNET（一个以能量最优为目标，另一个采用 $\epsilon = 0.5$），并在Bebop无人机上进行飞行测试，所得轨迹如图15所示。在这两种情况下，网络都能够调整其轨迹。对于速度更快的G&CNET，由于其策略更难学习（第III节），从航点产生的位置误差明显更大；此外，当飞行方式更加趋近时间最优时，现实差距（硬件延迟、状态估计误差和建模误差）也更难以应对。

这里也得出了与第IV节类似的观察结果，即

在更大范围的数据上训练 G&CNET 时，控制精度较低。我们使用经过训练、能够在一系列不同航点位置上飞行的 G&CNET，沿着采用固定航点的 4×3 米航迹飞行；与专门针对 4×3 米航迹训练的 G&CNET（图14）相比，该 G&CNET 飞行的圈更加不一致（图16）。两个网络的网络架构大致相同（在 $WP_{1、x}$ 和 $WP_{1、y}$ 均发生变化的情况下，仅多了一个输入神经元）。当前的网络规模可能需要扩大，才能使两个 G&CNET 以相同的精度飞行 4×3 米航迹。

VI. 结论

我们研究了制导与控制网络在四旋翼无人机高速飞行中的应用。我们证明，相较于能量最优控制问题，时间最优控制问题的控制策略要难学得多。对于 Bebop 1，要实现接近时间最优的飞行，平均控制误差 $\pm 3.35\%$ 在仿真中已高到足以妨碍稳定飞行。我们证明，螺旋桨的最大角速度ω_{max} 会影响时间最优控制轨迹中的切换时间。随后，我们进一步表明，对 ω_{max}的高估或低估程度越大，最优轨迹的平均位置误差就越大，进而影响飞行的鲁棒性。我们提出了一种峰值跟踪算法，用于机载识别 ω_{max} ，并结合 G&CNET，使其能够根据识别出的 ω_{max}数值调整控制策略。当四个旋翼中的一个达到饱和后，我们的算法只需 0.1s 即可识别新的限值；即使最初将 ω_{max} 高估了 $+700$ 转/分钟，也能在真实飞行中保持接近最优轨迹。最后，我们扩展了此前关于 G&CNET 的工作，使其能够在飞行时将前方两个航路点纳入考虑。新的流程能够生成包含特定机动动作的训练数据集，供 G&CNET 学习，例如使其能够在仿真中沿八字形赛道飞行。与单航路点飞行相比，由于多航路点的最优控制问题 formulation 更能代表整个控制任务，我们在一条 4×3米长的赛道上更好地优化了能量最优代价函数。我们研究了如何以尽可能快的速度飞完 4×3米长的赛道，并将 G&CNET 与最先进的微分平坦性控制器（DFBC）进行基准比较。我们证明，G&CNET 的赛道圈速与 DFBC 相近，并且能够适应不断变化的航点位置。这凸显了 G&CNET 相较于 DFBC 等其他基于最优性的方案的一项主要优势：它能够灵活、快速地重新计算最优控制输入。

未来的工作可以研究如何识别每个旋翼各自的最大角速度，从而避免因承受气动载荷最大的螺旋桨而限制承受较小气动载荷的螺旋桨。还可以实施更严格的约束，确保航路点始终位于摄像头的视场内。当前 Bebop 的动力学模型未纳入下洗流的影响；在.....的飞行中，推力与阻力模型也常有误差，因此，可以考虑将域随机化与机载测量结合起来，在飞行过程中适应这些模型误差。最后，本研究针对两个连续航路点所采用的方法，可用于训练 G&CNET，使其涵盖范围广得多的可能航点组合，从而让网络能够飞出更多不同的航迹。

致谢

作者谨此感谢 Emmanuel Blazquez 和 Alexander Hadjiivanov 在整个项目期间提供的宝贵意见和讨论，并感谢 Erik van der Horst 提供技术支持。本研究由欧洲空间局探索计划共同资助，并由欧洲空间局提供资金。

参考文献

[1] L. Bauersfeld 和 D. Scaramuzza，《多旋翼飞行器的航程、续航时间和最优速度估算》，2021年。 [在线]。获取地址：https://arxiv.org/abs/ 2109 04741 。[2] C. De Wagter、F. Paredes-Valles、N. Sheth 和 G. de Croon，《自主无人机竞速中的快速学习》，《自然·机器智能》，第3卷，第10期，第923页，2021年，版权：Copyright 2021 Elsevier B.V.，保留所有权利。[3] G. Loianno、C. Brunner、G. McGrath 和 V. Kumar，《利用单个摄像头和惯性测量单元对小型四旋翼进行激进飞行的状态估计、控制与规划》，《*IEEE*机器人与自动化快报》，第2卷，第2期，第404–411页，2017年。[4] A. Romero、S. Sun、P. Foehn 和 D. Scaramuzza，《时间最优四旋翼飞行的模型预测轮廓控制》，2021年。 [在线]。获取地址：https://arxiv.org/abs/ 2108 13205 。[5] K. Mohta、M. Watterson、Y. Mulgaonkar、S. Liu、C. Qu、A. Makineni、K. Saulnier、K. Sun、A. Zhu、J. Delmerico、K. Karydis、N. Atanasov、G. Loianno、D. Scaramuzza、K. Daniilidis、C. Taylor 和 V. Kumar，《在无 GPS 和复杂环境中实现快速自主飞行》，《野外机器人学期刊》，第35卷，2017年12月。[6] S. Li、M. M. Ozo、C. De Wagter 和 G. C. de Croon，《自主无人机竞速：一种计算效率高的基于视觉的导航与控制策略》，《机器人与自主系统》，第133卷，第103621页，2020年。 [在线]。获取地址：https: //www.sciencedirect.com/science/article/pii/S 0921889020304619 。[7] D. Mellinger、N. Michael 和 V. Kumar，《四旋翼精确激进机动的轨迹生成与控制》，《国际机器人研究杂志》，第31卷，第5期，第664–674页，2012年。 [在线]。获取地址：https://doi.org/10.1177/ 0278364911434236[8] E. Kaufmann、A. Loquercio、R. Ranftl、M. M¨uller、V. Koltun 和 D. Scaramuzza，《无人机高难度特技》，*RSS*：机器人、科学与系统，2020年。[9] P. Foehn、A. Romero 和 D. Scaramuzza，《四旋翼航点飞行的时间最优规划》，《科学·机器人学》，第6卷，第56期，第eabh1221页，2021年。 [在线]。获取地址：https://www.science.org/doi/abs/ 10.1126/scirobotics.abh1221[10] T. Salzmann、E. Kaufmann、M. Pavone、D. Scaramuzza 和 M. Ryll，《神经网络模型预测控制：面向四旋翼和敏捷机器人平台的深度学习模型预测控制》，2022年。 [在线]。获取地址：https://arxiv.org/abs/ 2203 07747 。[11] D. Mellinger 和 V. Kumar，《四旋翼的最小化加加速度轨迹生成与控制》，载于*2011*年*IEEE*国际机器人与自动化会议，2011年，第2520–2525页。[12] S. Li、E. Ozt¨urk、C. D. Wagter、G. C. H. E. de Croon 和 D. Izzo，《通过最优性原理的深度网络表示实现四旋翼激进在线控制》，*CoRR*，第abs/ 1912 07067, 卷。。在线。获取 2019 [ ] 1912 地址：http://arxiv.org/abs/ 。 07067 [13] R. Ferede、C. De Wagter、G. C. H. E. de Croon 和 D. Izzo，《基于神经网络的最优四旋翼控制器的自适应控制策略》，硕士论文，代尔夫特理工大学航空航天工程系，2022年。 [在线]。获取地址：http://resolver.tudelft.nl/uuid: b43a9703-082c-47c7-a56e-d50794ee8c1c

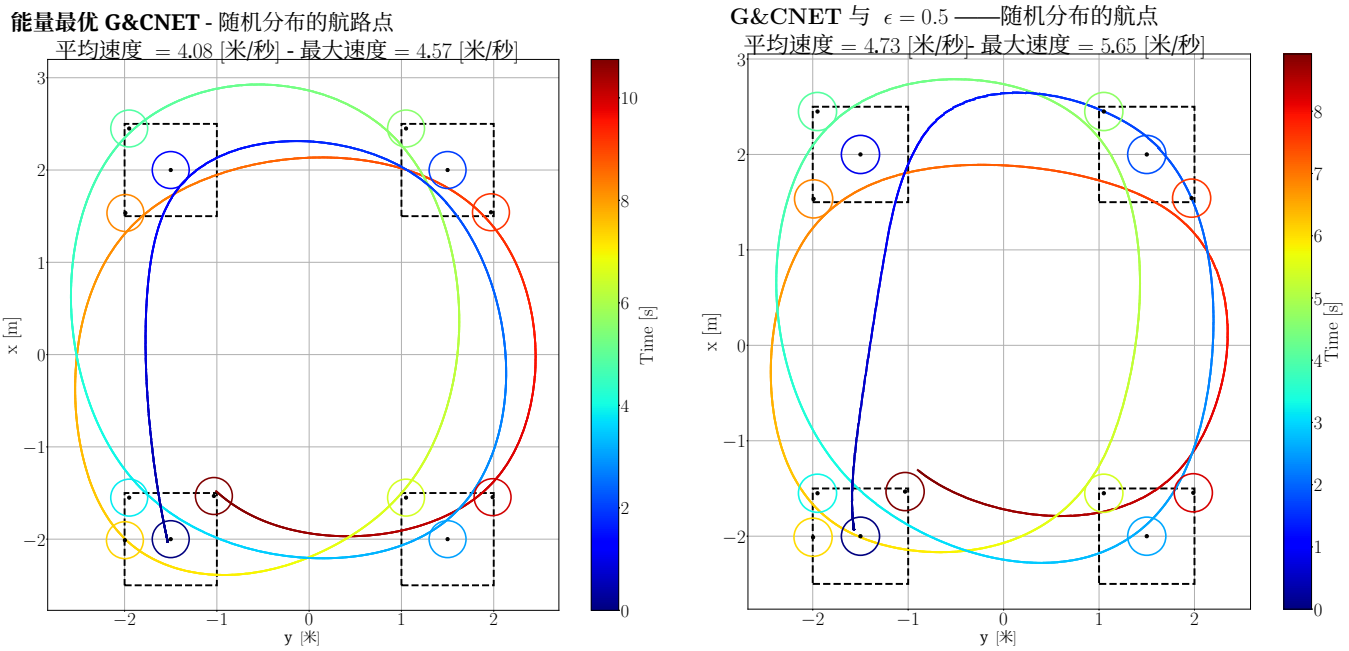

图 15. $\epsilon = 1.0$ （左）和 $\epsilon = 0.5$ （右）真实飞行的轨迹（俯视图）。G&CNET 在相对位置不断变化的 2 个连续航路点上进行训练。我们在虚线矩形内随机设置航路点。圆圈表示每个航路点的位置约束。

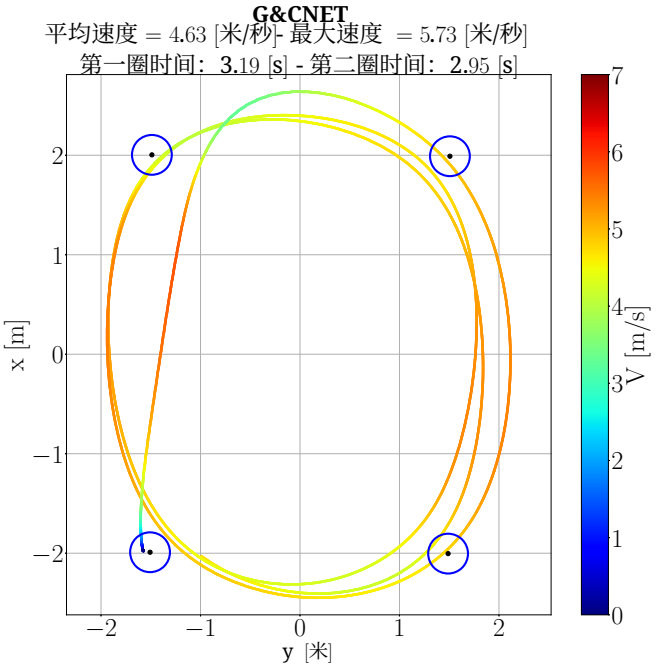

图 16. 真实飞行的轨迹（俯视图） $\epsilon = 0.5$。G&CNET 使用 2 个相对位置各异的连续航路点进行训练。蓝色圆圈表示中间航点的位置约束。

[14] C. S´anchez-S´anchez 和 D. Izzo，《通过深度神经网络实现实时最优控制：着陆问题研究》，《制导、控制与动力学杂志》，第41卷，2016年10月。[15] D. Izzo 和 E. ¨Ozt¨urk，《利用深度神经网络实现低推力转移的实时制导》，《制导、控制与动力学杂志》，第44卷，第2期，第315–327页，2021年。[16] D. Izzo 和 S. Origer，《时间最优恒加速度交会的神经表示》，《宇航学报》，2022年。[在线]。获取地址：https://www.sciencedirect.com/science/article/pii/S0094576522004581

[17] J. Svacha、K. Mohta 和 V. R. Kumar，《通过补偿空气动力学效应改进四旋翼航迹跟踪》，*2017*年国际无人机系统会议（*ICUAS*），第 860–866 页，2017 年。[18] S. Sun、C. C. de Visser 和 Q. Chu，《基于高速飞行数据的四旋翼灰箱模型辨识》，《航空器杂志》，第 56 卷，第 2 期，第 645–661 页，2019 年 3 月。 [在线]。网址：https://doi.org/10 2514 1 035135 。/ .c R. Fourer、D. M. Gay 和 B. [19] W. Kernighan，《用于数学规划的建模语言》，《管理科学》，第 36 卷，第 5 期，第 519–554 页，1990 年。[20] P. E. Gill、W. Murray 和 M. A. Saunders，《SNOPT：一种用于大规模约束优化的序列二次规划算法》，《*SIAM*评论》，第 47 卷，第 1 期，第 99–131 页，2005 年。[21] D. P. Kingma 和 J. Ba，《Adam：一种随机优化方法》，*arXiv*预印本 *arXiv:1412.6980*，2014 年。[22] B. Gati，《用于学术研究的开源自动驾驶仪——Paparazzi系统》，载于*2013*年美国控制会议，2013 年，第 1478–1481 页。[23] J. Dormand 和 P. Prince，《一类嵌入式龙格－库塔公式》，《计算与应用数学杂志》，第 6 卷，第 1 期，第 19–26 页，1980 年。 [在线]。网址：https: // www.sciencedirect.com/science/article/pii/0771050X80900133

VII. 附录

A. 更复杂航迹的示例

图17展示了使用学会飞行八字形的 G&CNET 进行仿真飞行的轨迹。

B. 电池对最大转速限制的影响

我们在 Bebop 上测试了一个带有 $\epsilon = 0.5$ 的 G&CNET，在一条 4×3 米航迹上飞行了 6 分钟。图18 是旋翼 4（见图 2）的放大图，由于飞行路径主要由右转弯组成，该旋翼大部分时间都处于饱和状态。随着电池电量耗尽，ω_{max} 约降低 1 转/分钟$^{-1}$。

C. G&CNET 训练的最优轨迹

图19展示了十条单航点及连续航点飞行的最优轨迹，这些轨迹可见于 G&CNET 的训练数据集中。

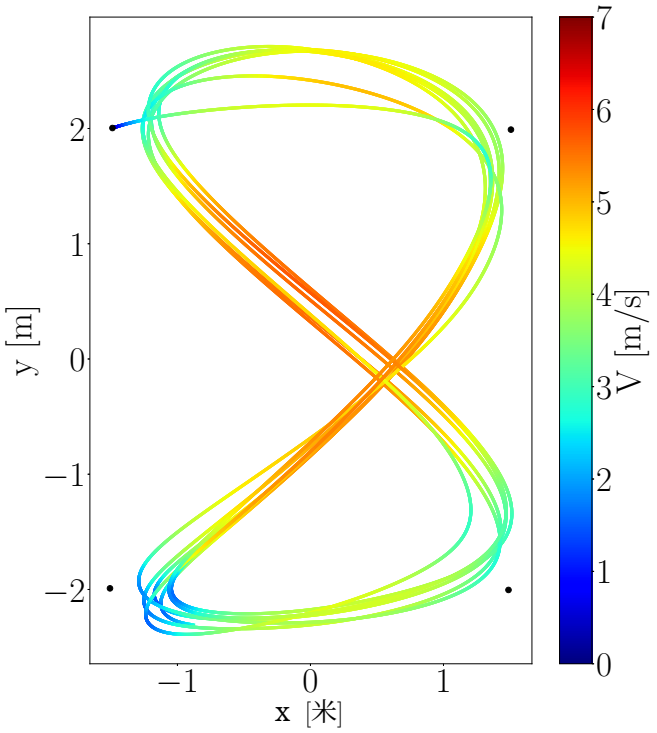

图17. 使用 $\epsilon = 0.4$ 在仿真中进行飞行的轨迹（俯视图）。G&CNET在4组急转弯数据上进行训练。飞行过程中，我们通过两个额外输入向G&CNET 提供即将到来的两个航路点的相对距离和角度信息。

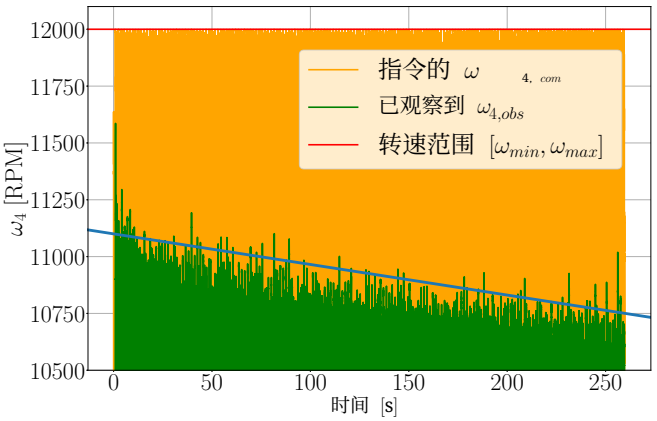

图18. 测试飞行期间指令角速度与观测角速度 ω，其中 $\epsilon = 0.5$（仅显示编号为4的旋翼）。蓝线大致表示 ω_{max} 的下降趋势。

D. 控制输入：*DFBC* 与 *G&CNET* 对比

图20展示了使用微分平坦性控制器（DFBC）和引导与控制网络（见第V节）进行真实飞行时，旋翼的指令角速度和观测角速度。

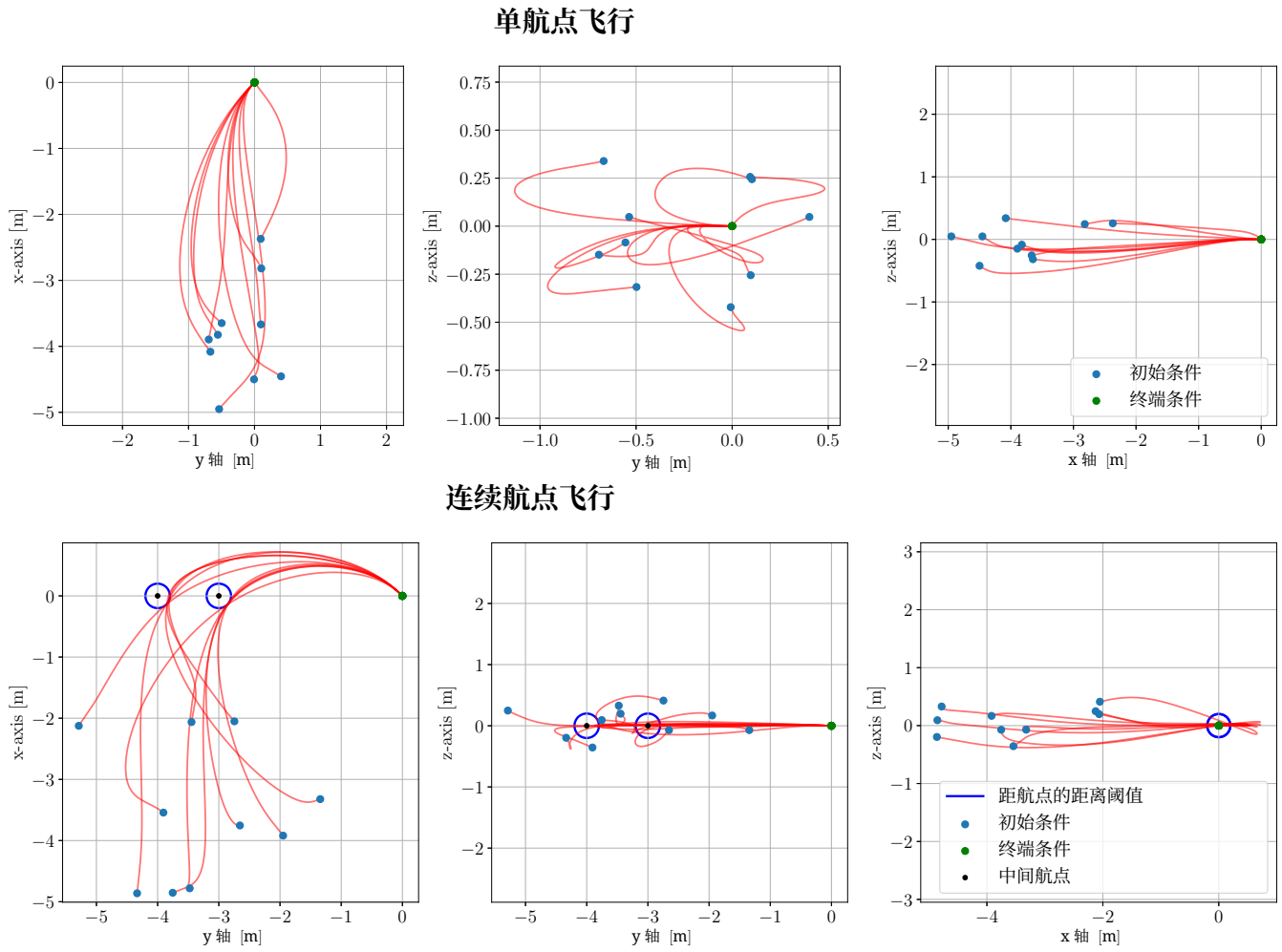

图19. G&CNET 训练数据集中单航路点与连续航路点飞行的十条最优轨迹。

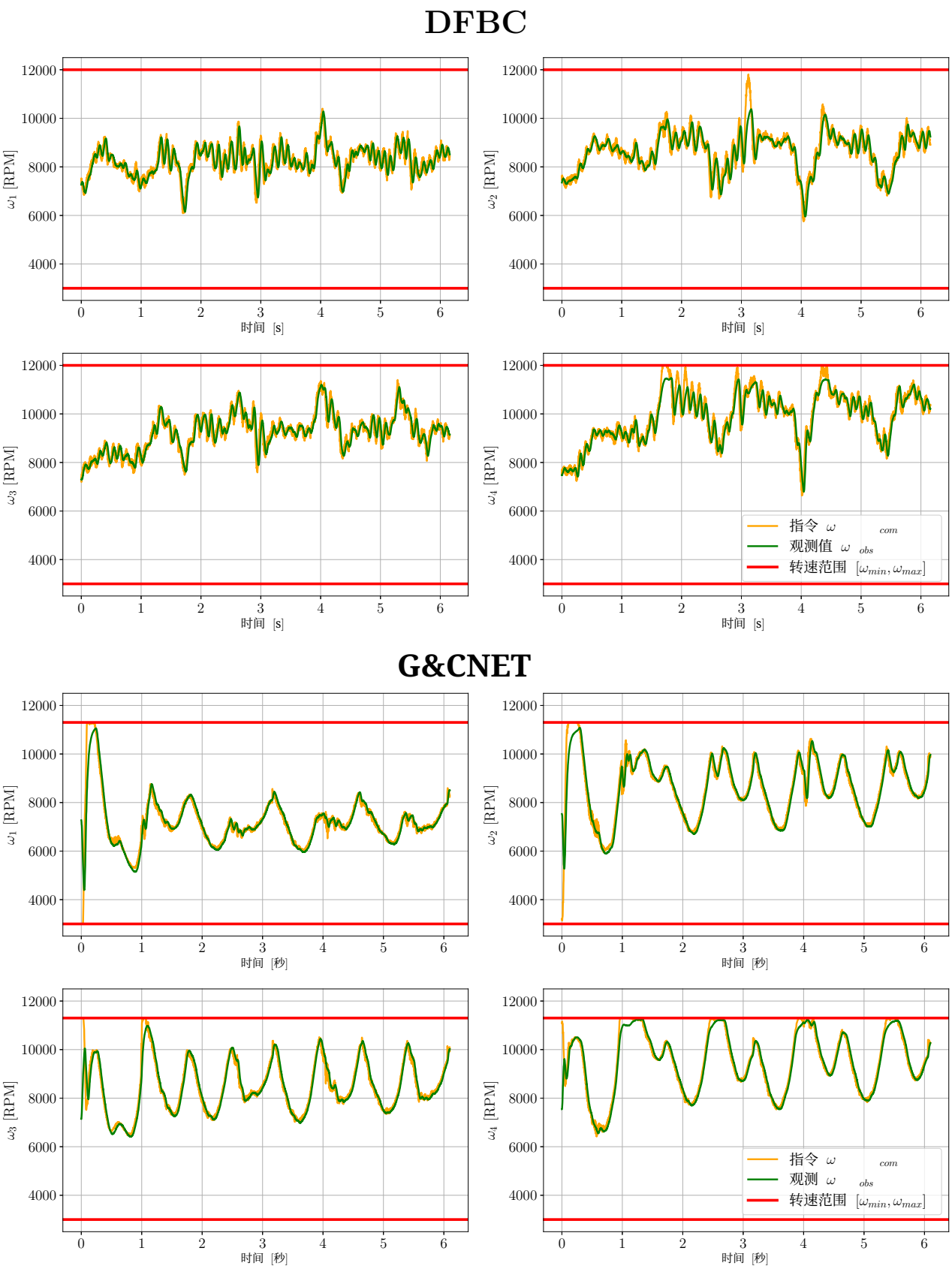

图 20. SecV 中 DFBC 和 G&CNET 的旋翼指令角速度与观测角速度。