ATMO: An Aerially Transforming Morphobot for Dynamic Ground Aerial Transition: Supplementary Notes

Ioannis Mandralis, Reza Nemovi, Alireza Ramezani, Richard M. Murray, Morteza Gharib

Supplementary Note 1 - Comparison to Cascaded PID Controller

We compare the ground-aerial morpho-transition controller presented in the main manuscript to a standard cascaded proportional, integral, derivative (PID) controller. In particular, we consider an open source implementation of the PX4 [1] flight controller - a flight controller which is widely adopted by the drone community. The PX4 controller relies on a cascaded control structure with a proportional position controller which produces a velocity reference to a PD velocity controller, which in turn produces acceleration and angular setpoints which are passed in to an SE3 attitude controller [2]. Finally, the attitude controller produces body rate setpoints for a PID body rate controller which produces the final thruster pulse-width modulation signals. The exact implementation we used can be found at the following link: https://github.com/PX4/PX4-Autopilot.

The controllers are compared in the Gazebo simulation environment [3] through numerical experiments. A real-world comparison cannot be conducted since, as will be seen below, the cascaded PID controller leads to dangerous instability. The robot was modeled using the identified dynamic parameters (see Supplementary Note 3), and each controller was tasked with sending the control commands subject to a given task.

We commanded a reference trajectory which consisted of flying up to a 5 meter altitude and following a square 4x4 meter path. We first considered the case without any mid-flight change of tilt angle, to validate both our controller and the cascaded PID controller. We used the auto-tune functionality of PX4 to select the cascaded PID control parameters. The results are shown in Figure S1(A). Both controllers are able to successfully track the reference trajectory. Our MPC controller has overall better tracking performance with minimal overshoot, but the cascaded PID controller could also potentially improve with better tuning parameter selection.

Now we consider the case where both controllers are tasked with following the same square trajectory, but with the tilt angle now going from 0 to 50 degrees along the first edge of the square path. The results are shown in Figure S1(B). Again our controller is able to track the reference trajectory even while morphing mid-flight. In contrast, the PX4 flight controller goes unstable and ultimately crashes into the ground. The reader is referred to supplementary video 7 for the Gazebo simulations that produced these results.

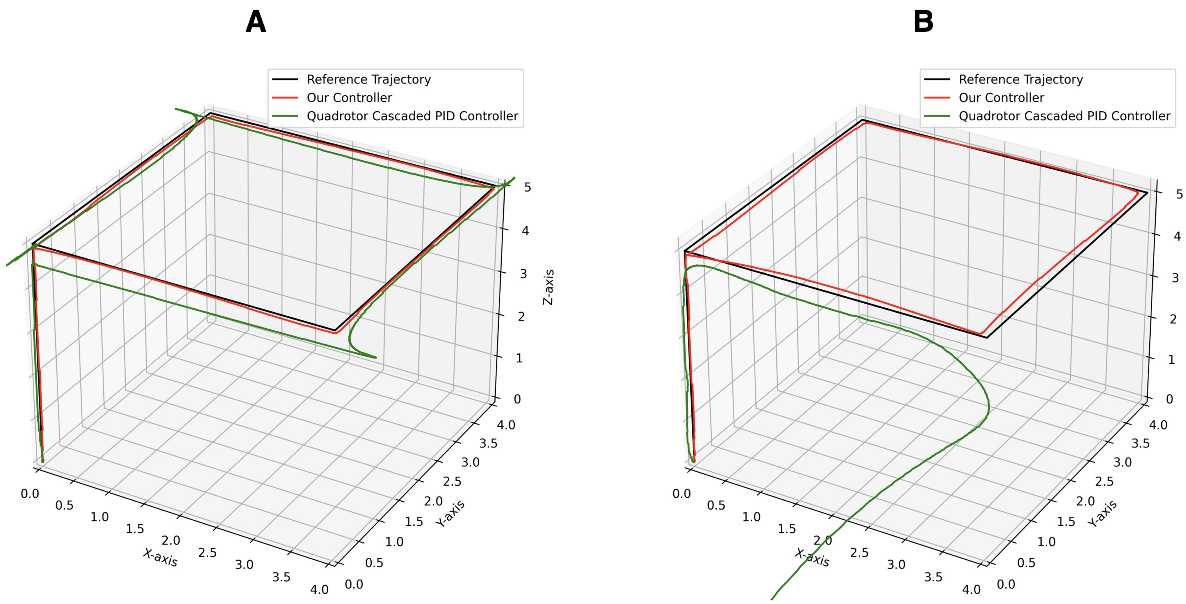

Figure 1: **Comparison to Cascaded PID Controller.** (**A**) Tracking performance with no mid-flight change in tilt angle. The reference trajectory (black) is compared to the trajectory achieved by our controller (red) and the PX4 Cascaded PID Controller (green). Our controller exhibits better performance with no overshoot and very close tracking. The standard PX4 controller has reasonable tracking performance with small overshoots. (**B**) Tracking performance subject to mid-flight change in tilt angle. Now the tilt angle starts to change as soon as ATMO reaches the first way point at 5 meters altitude. Our controller is still able to achieve adequate tracking performance whilst the PX4 flight controller goes unstable. See Video 7 for a video of the Gazebo simulations that produced these results.

The cascaded PID controller is able to control ATMO when the tilt angle is zero since, in this case, ATMO is equivalent to a quadrotor. However, as ATMO transforms mid-flight, the cascaded PID controller can no longer stabilize the system. This is due to new dynamic couplings which fundamentally change the system dynamics and render the cascaded control architecture developed for quadrotors inapplicable.

An example of the forces which occur during morphing flight are depicted in Figure S2(A), where T denotes the overall thrust force and φ is the tilt angle. Here, one can see that when the robot is mid-flight in a tilted configuration, roll motion and horizontal motion are dynamically coupled. If a roll torque is applied to the robot by applying more thrust on the left side than the right, an unwanted horizontal thrust force is also created in the opposite direction. Indeed, if the robot attempts to roll in order to translate, this results in a horizontal component of the thrust force of magnitude $T \sin \varphi$ in the undesired direction. Since quadrotor controllers need to roll in order to translate in the same lateral direction, they become inapplicable under this morphed flight configuration.

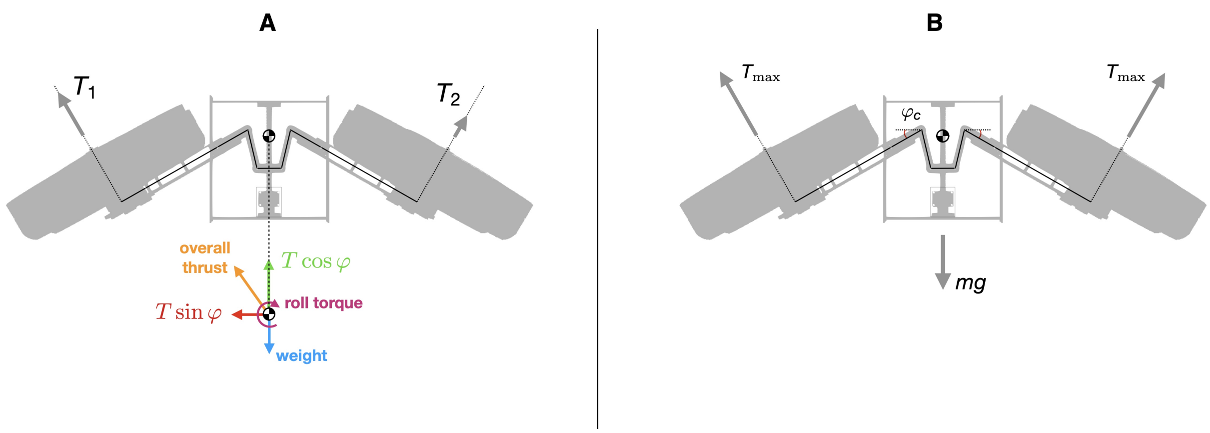

Figure 2: **Morphing Flight Dynamics.** The key kinematics parameter is the tilt angle φ which varies from $\varphi = 0$ in flight configuration to $\varphi = \frac{\pi}{2}$ in drive configuration. (**A**) Robot depicted in flight with morphing mechanism visible, and tilt angle φ. A free body diagram depicting the forces is shown. Because of the left and right thrust imbalance an unwanted horizontal thrust component is observed. The overall thrust points away from the vertical axis, making these dynamics fundamentally different to those of a quadrotor. (**B**) Robot depicted in critical angle configuration where $\cos\varphi_c = \frac{mg}{T_{\max}}$ and $T_{\max}$ is the sum of the maximum thrust of each thruster.

Supplementary Note 2 - Aerodynamics

In Figure 3 of the paper text the ground effect was reported for different tilt angles. Using the same aerodynamic test rig we measured how the thrust varies across different tilt angles away from the ground. In Figure 5 of these Supplementary Notes we report thrust values, normalized by the thrust at zero tilt angle i.e. $\frac{T(\varphi)}{T(\varphi=0)}$ to show how the thrust varies with tilt angle at a distance sufficiently far from the ground. We also plot the expected decay from rigid body mechanics which is $\frac{T(\varphi)}{T(\varphi=0)} = \cos\varphi$. This is the expected relationship between thrust and tilt angle if the thrust where pure force generators and all aerodynamic effects like rotor-rotor downwash interactions were ignored. As seen in Figure 5 below, the measured thrust values match the expected cosine decay until 50 degrees tilt and past that they begin to deviate slightly. This might indeed be due to rotor-rotor aerodynamic interactions, warranting further investigation.

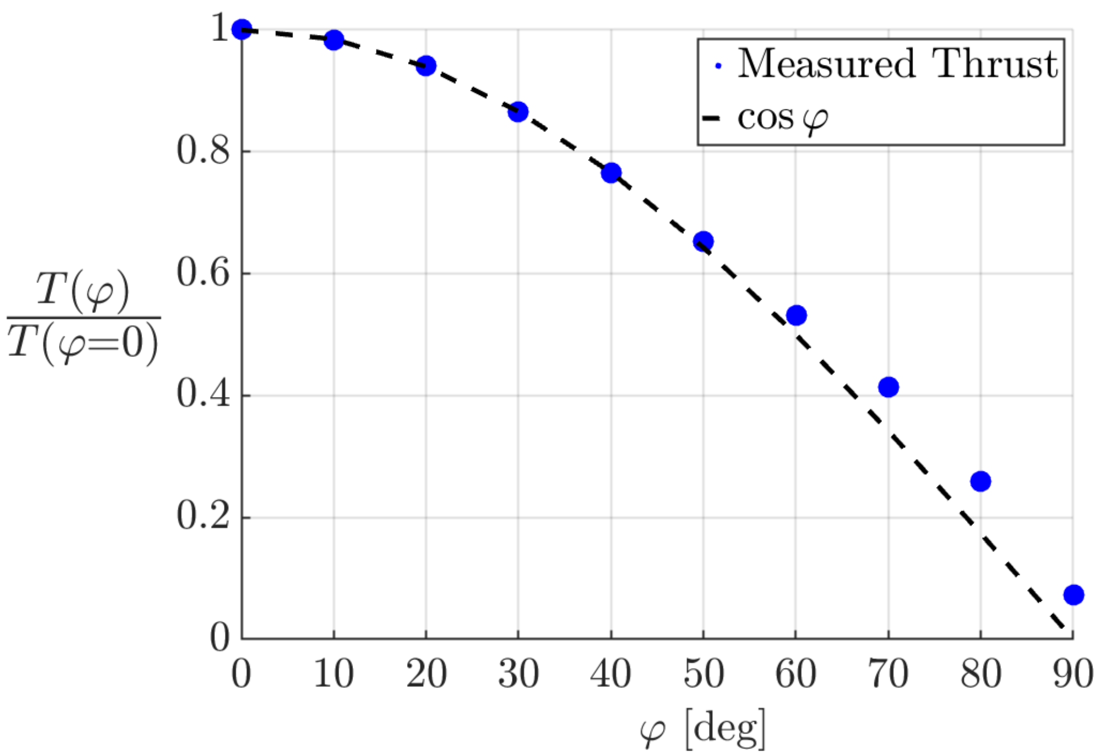

Figure 3: **Effect of tilt angle on thrust, away from ground.** The thrust at tilt angles $\varphi = 0, 10, 20, ..., 90°$, normalized by the thrust at zero tilt, is plotted against the expected theoretical decay of $\cos\varphi$.

The unnormalized values of the thrust as a function of the distance to the ground for different tilt angles, corresponding to those reported in Figure 3 of the manuscript are also reported in Figure S.4, for completeness.

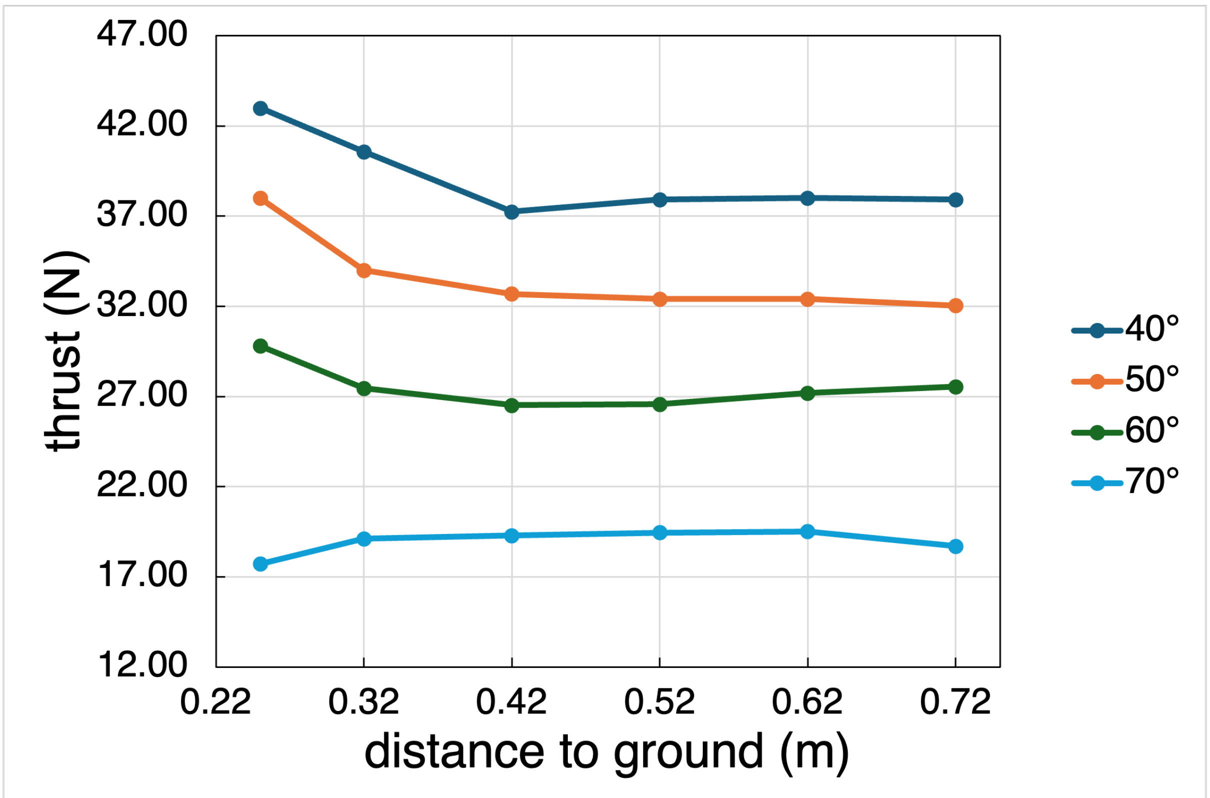

Figure 4: **Unnormalized thrust values as a function of distance to ground and tilt angle.** The nominal thrust at tilt angles $\varphi = 40, 50, 60, 70°$ is plotted against the distance to the ground.

Supplementary Note 3 - ATMO Detailed Hardware Description

The problem of combining ground and aerial locomotion in a single dynamically transforming robotic platform poses significant mechanical design challenges. The additional actuation increases the system weight, whilst the need to transform in the air requires sufficient torque to continually fight against the thrust forces. In order to keep the vehicle weight low enough for flight, the robot chassis, arms, and wheel components were fabricated using 2-mm carbon fiber plates and 3D printed parts. We used a fiber-inlay process on a MarkForged 3D printer with Onyx thermoplastic materials due to their excellent strength-to-weight ratios and ease of use, facilitating rapid prototyping and robot production. In the following, the major subsystems of the robot are described.

Wheels and wheel drive. The wheels are 3D printed and consist of a hollow cylinder which is embossed with narrow grooves to increase ground traction. The 3D printed frame is reinforced by thin rings of carbon fiber for additional rigidity. There are two brushed DC motors which drive the wheels on either side of the robot using a belt mechanism that rotates both wheels simultaneously allowing for differential drive steering. Driving is performed by direct pulse-width-modulation (PWM) control of the DC motor on either side of the robot using a radio controller or ROS2 control node.

Propellers and propulsion system. The wheel-thruster actuators contain the thruster and propeller which rotate independently of the wheel. The wheel acts as a shroud around the propeller for protection. There are potential performance benefits (noise reduction, aerodynamic efficiency) to be gained by such a shrouded propeller mechanism [4] but characterizing and optimizing these have been left for future work. We use 9-inch three-blade propellers, and a 6S brushless motor combination that can produce 2.9kg of thrust, resulting in a lift-to-weight ratio of approximately 2 and significant payload capacity. The motors are controlled using an onboard CubeOrange flight controller running the PX4 firmware [1] which sends PWM signals to each brushless motor based on user input from the radio controller or the offboard ROS2 control node.

Morphing mechanism. The morphing mechanism consists of a brushed DC motor that spins a worm gear which in turn tilts both the left and right arms of the robot simultaneously. The reader is referred to S1 for a detailed depiction of the mechanism. Because of the mechanism's gear reduction the torque required of the DC motor is relatively low, allowing for a lightweight and cheap actuator to drive the system. Furthermore, the tilting is symmetric and self-locking reducing the mechanical power required by the tilting motor. Note that using servo-motors instead of a custom tilt mechanism is possible but would likely require much heavier and more expensive actuators to deliver the torque required to transform the robot in flight.

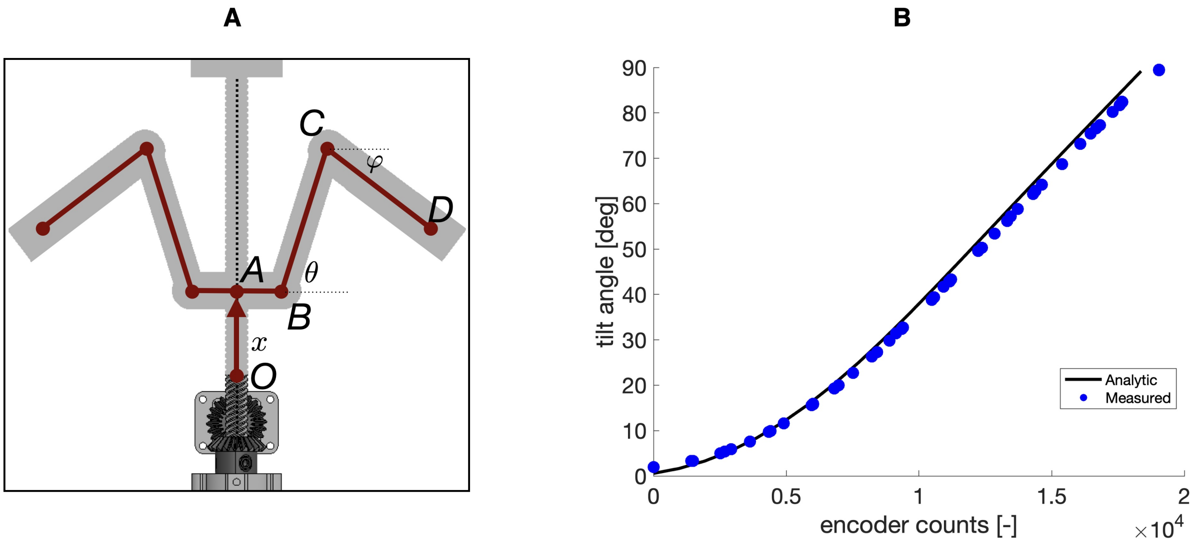

Figure 5: **Morphing kinematics and validation.** (**A**) A closeup schematic of the tilt actuator and all relevant variables. θ, φ are the internal angles of the mechanism and x is the displacement of A with respect to O. (**B**) An experimental validation of the closed loop kinematic equations which predict the tilt angle φ (in solid black) against the number of encoder counts of the actual motor (blue markers).

Electronics. The electronics stack is based on a central onboard computer, the NVIDIA Jetson Orin Nano. This uses the Robot Operating System 2 (ROS2) [5] to communicate with the motor drivers actuating the tilt actuator and driving motors, as well as the flight controller, and the electronic speed controllers (ESCs) which actuate the thruster-propeller pairs. ROS2 enables running a central, real-time, control loop that receives inputs from an RC transmitter/receiver pair, as well as pose (3D position and orientation) data from a motion-capture ground computer running the Optitrack Motive software. With this information the central control loop does the necessary computations and sends signals to the motor drivers through ROS2, which in turn send pulse-width modulation signals (PWM) to the respective actuators. More specifically, the pose information being streamed by the Motion Capture ground computer is sent to the CubeOrange flight controller (flashed with a custom version of the PX4 autopilot firmware) via the UXRCE-DDS protocol which translates ROS2 messages to PX4 compatible uORB messages. This pose data is given as an input to the Extended Kalman Filter (EKF) which runs onboard the CubeOrange. This fuses the Motion Capture pose information with the onboard IMU information, as well as other available sensor information (magnetometer, barometer, etc.) to provide a real-time state estimate. The PX4 EKF runs a delayed fusion time horizon EKF to allow for different time delays relative to the IMU.

System identification and simulation. In order to obtain good estimates of the performance characteristics of the propeller-thruster combination, we performed bench tests using a 3-axis load cell and the software RC Benchmark. This resulted in a thrust coefficient $k_T = 28.015$ and a moment coefficient

$k_M = 0.018$. To obtain the mass and inertial parameters the 3D model of the robot design (CAD) was used. The reader is referred to Tables S3 for all the parameters resulting from the system identification. Using the identified parameters, the controller was extensively tested in simulation before deployment in the real world. We used the rigid body dynamics simulator Gazebo [3] coupled with PX4 software-in-the-loop. The simulation was used to test the robustness of the controller by adding measurement noise to our state estimates, actuator noise, as well as random force and torque disturbances, before conducting experiments.

Onboard computer	NVIDIA Jetson Orin Nano Developer Kit
Flight Controller	CubeOrange+
RC transmitter/receiver	Futaba T18SZ 18 channel/R7008SB Futaba
Tilt/Wheel actuator drivers	Roboclaw 2x15A motor controller
ESCs	APD 80F3[X]v2
Thrusters/propellers	T-motor Cine series 1155 Kv/HQ 9x5x5
Tilt actuator	Pololu 12V high power 47:1 (4845)
Wheel actuators	Servocity Econ spur gear motors 170RPM
Battery pack	Hoovo 6s 8000 mAh 100C

Figure 6: **Electronic components list.** A list of all the main electronic components used for the onboard control, communication, and actuation.

A

k_T	28.015
k_M	$\pm$ 0.018

B

m_{base}	2.33 kg
m_{arm}	1.537 kg
m_{rotor}	0.021 kg
m_{total}	5.49 kg

C

I^{xx}_{base}	0.0067
I^{xy}_{base}	-0.000031
I^{xz}_{base}	0.00046
I^{yy}_{base}	0.011
I^{yz}_{base}	0.00004
I^{zz}_{base}	0.0088
I^{xx}_{arm}	0.008732
I^{xy}_{arm}	0.000007
I^{xz}_{arm}	-0.000012
I^{yy}_{arm}	0.036926
I^{yz}_{arm}	0.000571
I^{zz}_{arm}	0.043822
I^{xx}_{rotor}	0.000022
I^{yy}_{rotor}	0.000022
I^{zz}_{rotor}	0.000043

Figure 7: **Aerodynamic, mass, and inertial parameters using for dynamics.** (**A**) Aerodynamics parameters identified by thrust bench tests. (**B**) Mass parameters identified from CAD model. (**C**) Inertial parameters identified from CAD model.

References

[1] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6235–6240, 2015.

[2] F. Goodarzi, D. Lee, and T. Lee, "Geometric nonlinear pid control of a quadrotor uav on se(3)," in *2013 European Control Conference (ECC)*, pp. 3845–3850, 2013.

[3] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, pp. 2149–2154 vol.3, 2004.

[4] E. K. Oshima, *Experimental Studies of Flow Control Techniques for Future Aircraft.* California Institute of Technology, 2023.

[5] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, May 2022.