2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

From CAD to URDF: Co-Design of a Jet-Powered Humanoid Robot Including CAD Geometry

Punith Reddy Vanteddu[1,2], Gabriele Nava[1], Fabio Bergonti[1], Giuseppe L'Erario[1], Antonello Paolino[1,3], Daniele Pucci[1,2]

***Abstract*— Co-design optimization strategies usually rely on simplified robot models extracted from CAD. While these models are useful for optimizing geometrical and inertial parameters for robot control, they might overlook important details essential for prototyping the optimized mechanical design. For instance, they may not account for mechanical stresses exerted on the optimized geometries and the complexity of assembly-level design. In this paper, we introduce a co-design framework aimed at improving both the control performance and mechanical design of our robot. Specifically, we identify the robot links that significantly influence control performance. The geometric characteristics of these links are parameterized and optimized using a multi-objective evolutionary algorithm to achieve optimal control performance. Additionally, an automated Finite Element Method (FEM) analysis is integrated into the framework to filter solutions not satisfying the required structural safety margin. We validate the framework by applying it to enhance the mechanical design for flight performance of the jet-powered humanoid robot iRonCub.**

I. Introduction

New frontiers in robotics research are investigating the connections between robots' cognitive and physical capabilities. This concept, known as *embodied intelligence*, leverages the idea that these capabilities should be addressed collectively rather than individually [1], [2]. Co-optimization and co-design emphasize the interdependence of these factors, acknowledging that improving one can influence the other, ultimately resulting in enhanced physical and cognitive abilities of the robot [3], [4], [5]. In the context of co-design, having a model of the robot is crucial as it serves as a starting point for the optimization process. However, the model commonly utilized is a simplified version of a complete CAD model, such as the Unified Robot Description Format (URDF) model [6], [7]. This representation, even though accurate for robot kinematics and dynamics, often lacks essential information required for prototyping the mechanical design of the optimized robot. As a result, the optimized model may not be immediately feasible, necessitating manual modifications and deviations during the prototyping phase, which lead to a suboptimal design. To overcome these limitations, we propose a co-design framework that uses both CAD and URDF models to optimize the jet interfaces of a flying humanoid robot.

[1]Artificial and Mechanical Intelligence, Istituto Italiano di Tecnologia, Genoa, Italy firstname.surname@iit.it
[2]School of Computer Science, University of Manchester, Manchester, UK
[3]Department of Industrial Engineering, University of Naples Federico II, Naples, Italy

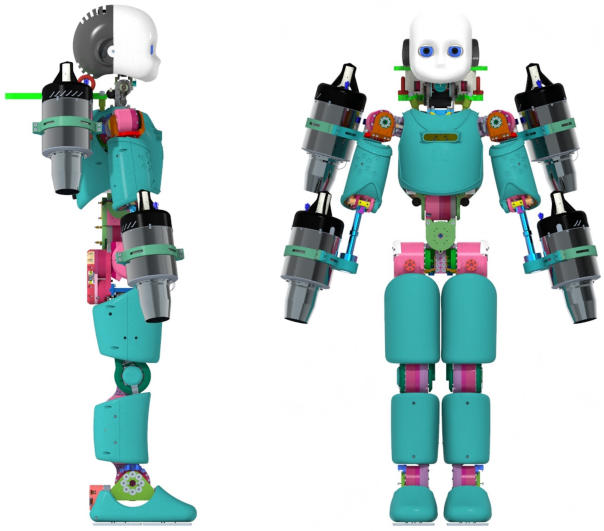


Fig. 1: iRonCub-Mk3 CAD model

In the field of terrestrial robotics, co-optimization often involves parameterizing robot design to optimize it for specific control-related objectives [8], [9]. This process may employ evolutionary algorithms to evaluate a wide range of populations and obtain the optimal design solutions [10], [11]. Integrating simulations of robot behavior into these optimizations not only aids in visualizing these behaviors but also expedites the overall design iteration process [12]. Such frameworks are useful for evaluating and comprehending the effectiveness of overall optimization strategies.

In literature, there are co-design frameworks integrating shapes, materials, and physics into the design process [13]. Task-based optimization techniques such as RoboGrammar enable the creation and optimization of diverse robot parts for specific tasks while adhering to CAD assembly rules [14]. Learning methods can also be utilized to understand trade-offs between different optimizations, enhancing robot performance across various terrains [15]. These evolutionary frameworks may yield new designs that outperform existing commercial solutions [16], [17]. However, optimization often relies on simple primitive CAD shapes, and applying these techniques to humanoids may prove overly simplistic given the complexity of humanoid parts and the scale of the optimization [18].

The development of an end-to-end framework integrating CAD model generation has been investigated [19]. However,

this approach does not ensure the optimal solution will meet the necessary safety factors, which is fundamental when optimizing the interfaces of a jet-powered humanoid robot. An alternative technique is discussed in another study [20], employing a cascaded optimization process where non-intuitive designs are eliminated in favor of selecting the most suitable design. However, in this scenario, a pre-existing pool of feasible solution space is constructed by examining of the shelf designs. In the use case of a humanoid, this space is thin and needs to be enhanced for better search.

Some inspiration can be obtained from robotic design optimization that has focused on improving the mechanical structures and drive-trains of robots using NSGA-II and other multi-objective algorithms to tackle the complex task of selecting motors, gearboxes, and link thicknesses for manipulators, balancing dynamic performance, structural deflections, and weight [21]. These methods have also been applied to optimize robot grippers, solving non-linear, multi-constraint challenges [22]. Similarly, topology optimization has helped design compliant quadruped legs, balancing stiffness and flexibility while offering Pareto-optimal solutions suitable for various robotic applications [23]

This paper introduces a co-design framework aimed at optimizing CAD parts of the jet-powered humanoid robot iRonCub [24], [25]. Specifically, CAD parts concerning the interfaces between the robot and the jets are parameterized in PTC-CREO. These parameterized CAD parts are then employed to derive optimal solutions using a constrained multi-objective evolutionary algorithm, namely NSGA-II [26]. More specifically, each population member is first assessed with a FEM analysis to ensure that the jet supports respect safety margins for failure due to stresses against external forces. Then, URDF models are generated, and the individuals are evaluated by computing the fitness function based on flight simulations using previously optimized flight controllers[27], [28]. The entire framework is automated and executed entirely through modeFRONTIER software.

The remainder of this paper is structured as follows: Section II recalls notation, robot modeling, and the flight control strategy; Section III details the implemented co-design framework; Section IV presents the results, including validation with aggressive trajectories to explore the robustness of the framework; Section V concludes the paper with final remarks and future directions.

II. Background

A. Notation

- $S(x) \in \mathbb{R}^{3\times3}$ is the skew-symmetric matrix associated with the cross product in $\mathbb{R}^3$, namely, given $x, y \in \mathbb{R}^3$, then $x \times y = S(x)y$.
- Vectors e_1, e_2, and e_3 are the canonical basis of $\mathbb{R}^3$.
- $\mathrm{SO}(3) := \left\{R \in \mathbb{R}^{3\times3} \,|\, R^\top R = I_3,\, \det(R) = 1\right\}$.
- $\mathcal{I}$ is the inertial frame, B denotes a frame rigidly attached to the robot base link, namely *base frame*.
- $\mathcal{G}[\mathcal{I}]$ is a frame with the origin at the robot center of mass $\mathcal{G}$ and the orientation of the inertial frame $\mathcal{I}$.
- $R_{\mathcal{B}} \in \mathrm{SO}(3)$ is the rotation matrix that transforms a 3D vector expressed with the orientation of the frame $\mathcal{B}$ in a 3D vector expressed in the frame $\mathcal{I}$.
- ${}^{\mathcal{G}[\mathcal{I}]}h \in \mathbb{R}^6$ is the momentum of the system expressed w.r.t. $\mathcal{G}[\mathcal{I}]$, namely centroidal momentum. ${}^{\mathcal{G}[\mathcal{I}]}h$ is defined as ${}^{\mathcal{G}[\mathcal{I}]}h = [{}^{\mathcal{G}[\mathcal{I}]}l; {}^{\mathcal{G}[\mathcal{I}]}w]$, with ${}^{\mathcal{G}[\mathcal{I}]}l, {}^{\mathcal{G}[\mathcal{I}]}w \in \mathbb{R}^3$ are the linear and the angular momentum.
- ${}^{\mathcal{G}[\mathcal{B}]}w = R_{\mathcal{B}}{}^{\mathcal{G}[\mathcal{I}]}w$ is the angular momentum with orientation represented in body coordinates.

B. Robot Modelling

The flying humanoid robot is modeled as a floating multi-body system composed of $n+1$ rigid bodies (links), n pin joints with 1 DoF each, and n_p thrusters rigidly mounted on the robot links. The robot configuration q is defined as $q := (p_B, R_B, s) \in \mathbb{R}^3\times SO(3)\times\mathbb{R}^n$. $p_B \in \mathbb{R}^3$ is the base position w.r.t. I, $R_B \in SO(3)$ is the base orientation w.r.t. $\mathcal{I}$, and $s \in \mathbb{R}^n$ is the vector of joint positions. The configuration velocity ν is defined as $\nu := (\dot{p}_B, \omega_B, \dot{s}) \in \mathbb{R}^3\times\mathbb{R}^3\times\mathbb{R}^n$. The base linear and angular velocities expressed in $\mathcal{I}$ are $\dot{p}_B, \omega_B \in \mathbb{R}^3$, and $\dot{s} \in \mathbb{R}^n$ is the vector of joint velocities.

C. Momentum-based Flight Controller

We recall the momentum-based flight controller designed in [24]. The rate of change of centroidal momentum is the sum of all the external forces acting in the body:

$$ {}^{\mathcal{G}[\mathcal{I}]}\dot{h} = \begin{bmatrix} {}^{\mathcal{G}[\mathcal{I}]}\dot{l} \\ {}^{\mathcal{G}[\mathcal{I}]}\dot{w} \end{bmatrix} = \begin{bmatrix} mge_3 + \sum_{k=1}^{n_p} F_k \\ \sum_{k=1}^{n_p} S(r_k)F_k \end{bmatrix}, \tag{1} $$

where $F_k = {}^{\mathcal{I}}a_k(q)T_k$ is the $k-th$ jet force composed of its direction ${}^{\mathcal{I}}a_k \in \mathbb{R}^{3\times3}$ and the thrust intensity $T_k \in \mathbb{R}^+$. Because of the slow thrust dynamics, we cannot solely control the thrust intensity to stabilize robot momentum: we need to control also the joint positions to modify jets direction via ${}^{\mathcal{I}}a_k(q)$. By applying *relative degree augmentation* to (1) with linear momentum in *centroidal coordinates* and angular momentum in *body coordinates*, we have:

$$ \ddot{h} := \begin{bmatrix} {}^{\mathcal{G}[\mathcal{I}]}\ddot{l} \\ {}^{\mathcal{G}[\mathcal{B}]}\ddot{w} \end{bmatrix} = \begin{bmatrix} \frac{\partial({}^{\mathcal{G}[\mathcal{I}]}\dot{l})}{\partial T}\dot{T} + \frac{\partial({}^{\mathcal{G}[\mathcal{I}]}\dot{l})}{\partial q}\nu \\ \frac{\partial({}^{\mathcal{G}[\mathcal{B}]}\dot{w})}{\partial T}\dot{T} + \frac{\partial({}^{\mathcal{G}[\mathcal{B}]}\dot{w})}{\partial q}\nu \end{bmatrix}, \tag{2} $$

Recalling that ν contains the joints velocities, we can now use $u := (\dot{T}, \dot{s})$ as the control input of (2) as they appear linearly in the equations. The (desired) closed-loop system dynamics is defined as follows:

$$ \ddot{h}^* := \begin{bmatrix} {}^{\mathcal{G}[\mathcal{I}]}\ddot{l}^* := {}^{\mathcal{G}[\mathcal{I}]}\ddot{l}_d - K_D\tilde{\dot{l}} - K_P\tilde{l} - K_I\int_0^t \tilde{l}\,dt \\ {}^{\mathcal{G}[\mathcal{B}]}\ddot{w}^* \end{bmatrix}, \tag{3} $$

where l_d is the reference linear momentum and $\tilde{l}$ is the linear momentum error, namely $\tilde{l} := ({}^{\mathcal{G}[\mathcal{I}]}l - {}^{\mathcal{G}[\mathcal{I}]}l_d)$. K_P, K_D,and K_I are the positive definite gain matrices, while ${}^{\mathcal{G}[\mathcal{B}]}\ddot{w}^*$ is calculated with a dedicated attitude controller [27]. Finally, the control input that achieves the desired dynamics (3) is obtained by solving the following Quadratic Programming (QP) problem

$$u^* = \underset{u}{\operatorname{argmin}}(\lambda_1|\ddot{l} - \ddot{l}^*|^2 + \lambda_2|\ddot{w} - \ddot{w}^*|^2 + \lambda_3|\dot{s} - \dot{s}^*|^2) \quad (4)$$
$$\text{s.t.} \qquad u_{min} \leq u \leq u_{max} \quad ,$$

where $|\dot{s} - \dot{s}^*|$ is a postural task that resolves input redundancy. The controller is implemented in Simulink and available in a public repository in GitHub [29].

III. Co-Design Framework

In this Section we outline the proposed co-design framework, discussing each part in a dedicated subsection. The entire framework, depicted in Fig. 2, is executed within modeFRONTIER, exploiting its inherent NSGA-II algorithm.

A. CAD Geometry

We identified CAD components essential for robot control, specifically those linking the jets to the jetpack and robot arms for the flying humanoid robot. These components, along with their geometrical features selected for optimization, are illustrated in Fig. 3. Our primary goal is to understand the impact of these geometric parameters, called θ, on the overall flight performance while maintaining at least the safety factor of the original CAD design. The CAD geometry is parameterized with geometric parameters θ using CREO-Parametric to enable FEM analysis. Changes in θ affect mass (M), volume, and center of mass position (CoM), and inertia (I_{XX}, I_{YY}, I_{ZZ}) of the parameterized parts. The CAD geometry is saved in both STEP and STL formats, enabling the potential utilization of alternative modeling software for additional post-processing.

B. FEM Structural Analysis

Each generated design undergoes verification through static structural analysis to verify the stress distribution on each part under turbine forces. This analysis is conducted using the MATLAB PDE toolbox. In this analysis, an external load of $250\,\mathrm{N}$, corresponding to the maximum thrust provided by the jets [30], is uniformly distributed over the surface area in contact with the turbine. The surface area is modified based on modifications to the geometry, which consequently affects the safety factor when considering a different design. One limitation of the PDE toolbox is its lack of support for the analysis of CAD assemblies, whereas our jet interfaces consist of multiple CAD parts each. Therefore, to conduct the FEM analysis, the assembly of each jet interface has been unified into a single STL mesh.

The material of the parts is ERGAL aluminum, with a Young's modulus of $71.7\,\mathrm{GPa}$ and a yield strength $\sigma_{y,\mathrm{Ergal}}$=$462\,\mathrm{MPa}$. The safety factor SF is calculated as

$$\mathrm{SF} = \frac{\sigma_{y,\mathrm{Ergal}}}{\sigma_{\mathrm{MAX}}}\,, \quad (5)$$

where σ_{MAX} is the maximum Von Mises stress acting on the part. The focus on static FEM analysis was chosen as it offers a simpler case that still provides meaningful results. We have opted for a high safety factor to ensure the stresses developed are below the desired criterion. An example of stress evaluation on one of the generated design candidates is illustrated in Fig. 4.

C. URDF Generation

We generate an initial complete URDF model of the robot from CAD using a semi-automatic procedure starting from the model illustrated in Figure 1. Subsequently, we systematically modify this URDF using a custom MATLAB algorithm tailored for updating the specific sections of the URDF that involve parametrized parts, incorporating the updated geometry and inertial parameters. This script enables the generation of multiple copies of the original URDF file, that are used to simulate robot flight for evaluating each population member.

D. Flight Simulation

We simulate a flight envelope consisting of the following actions: take-off from the initial position, forward movement, descent, and rotation along the yaw axis. Snapshots of the flight envelope and the simulator are depicted in Fig. 5. The flight motion is designed to encompass a wide range of actions that the robot may be required to perform during flight.

E. Design Optimization with NSGA-II

To achieve a well-distributed sequence of points across the solution space and ensure comprehensive coverage, we initiate the optimization process by constructing a population of 25 candidates using a quasi-random distribution with the Sobol algorithm. Then, a genetic algorithm spanning 40 generations is utilized to refine these initial candidates by minimizing our selected cost functions. During the generation of a new population, the algorithm conducts structural analysis and eliminates infeasible designs. Additionally, we conduct a further check based on the status of the QP solver to eliminate solutions in which the solver failed to stabilize the robot. While it is feasible to also incorporate gains and other control parameters into the optimization process, doing so adds complexity and significantly extends the time required for convergence to optimal solutions. To preserve the simplicity of the problem and better understand the correlation between CAD-related parameters and flight performance, we have chosen not to include control-related parameters at this stage.

In the settings of the NSGA-II algorithm, the crossover probability parameter was specified as 0.9, and the mutation probability parameter was defined as 0.25. The geometric parameters are encoded in a vector of integers generated within specified bounds and with a specified step size, as reported in Table I.

The fitness functions minimized by the genetic algorithms are: i) the tracking of linear and angular momentum

$$\delta_h = \left|\sum_{i=1}^{n_t} \tilde{l}(t_i)\right|^2 + \left|\sum_{i=1}^{n_t} \tilde{w}(t_i)\right|^2$$

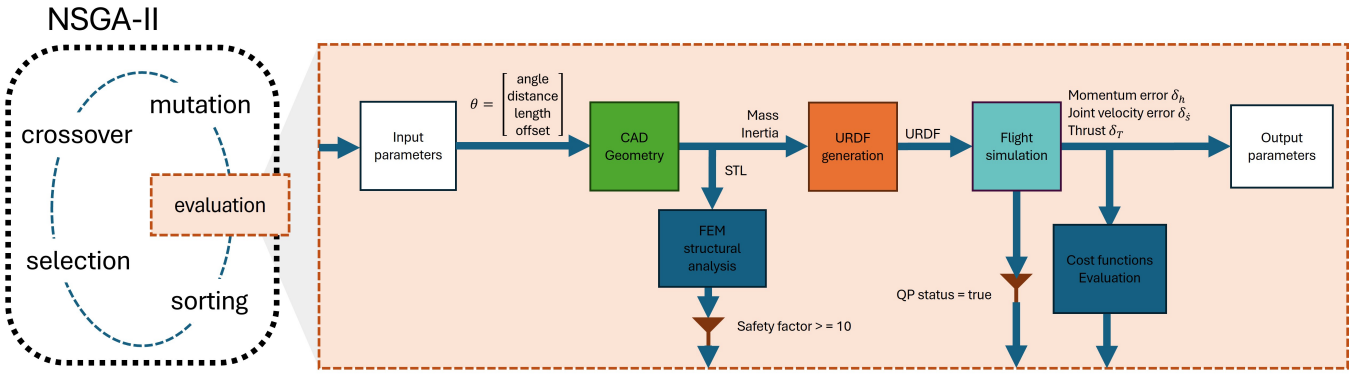


Fig. 2: Representation of the co-design framework and the individual evaluation strategy. The brown arrows represent the constraints that must be satisfied to obtain a feasible solution.

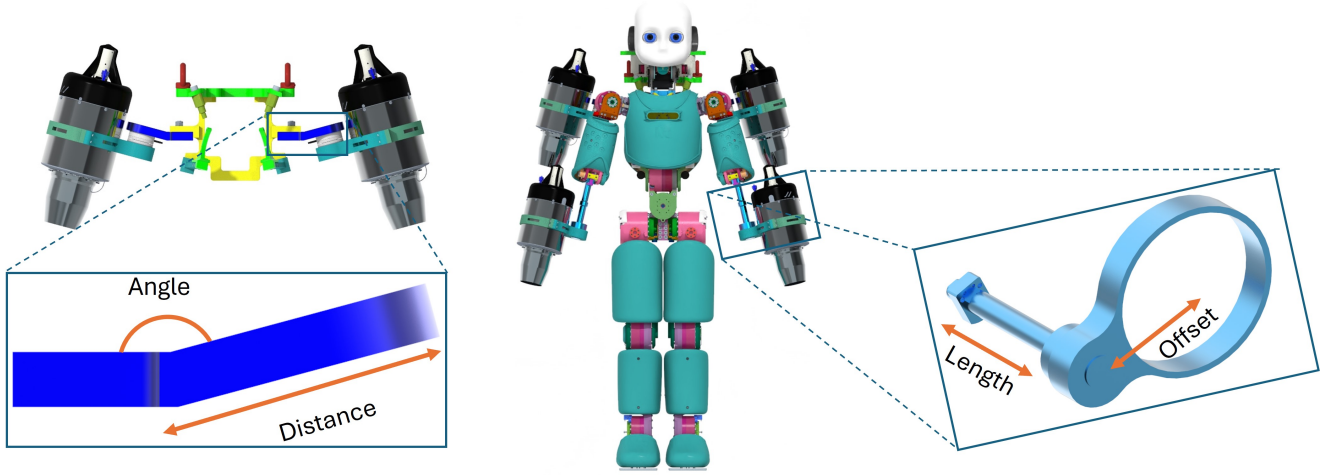


Fig. 3: Geometry of the components to be optimized for enhancing jet positions and orientations. The geometry parameters *Angle*, *Distance*, *Length*, and *Offset* are collected in a vector θ.

TABLE I: Bounds and step size for the geometric parameters of the optimization.

Parameter	min	max	step
Angle [°]	1	79	1
Distance [mm]	40	100	2
Offset [mm]	80	120	2
Length [mm]	50	150	2

ii) the joint velocity tracking of the torso and the left and right arms

$$\delta_{\dot{s}} = 2\left|\sum_{i=1}^{n_t}\left(\dot{s}_{\text{torso}} - \dot{s}^d_{\text{torso}}\right)\right|^2 + \left|\sum_{i=1}^{n_t}\left(\dot{s}_{\text{arms}} - \dot{s}^d_{\text{arms}}\right)\right|^2;$$

and *iii*) the time-averaged total thrust required to execute the whole trajectory

$$\delta_T = \sum_{i=1}^{n_t} \frac{T_1(t_i) + T_2(t_i) + T_3(t_i) + T_4(t_i)}{n_t},$$

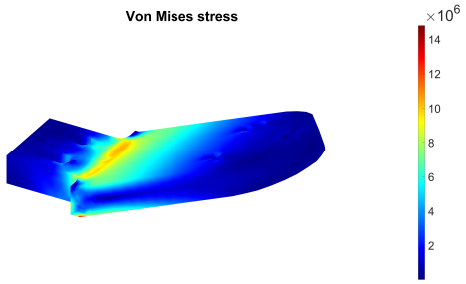
with n_t is the number of steps used to discretize the flight horizon. Augmenting the fitness functions with the constraints, we can formulate the multi-objective optimisation problem as:

$$\theta^* = \underset{\theta}{\text{argmin}} \begin{bmatrix} \delta_h \\ \delta_{\dot{s}} \\ \delta_T \end{bmatrix} \tag{6}$$
$$\text{s.t.} \quad \text{SF} \geq 10\,,$$
$$\text{QP}_{\text{status}} = \text{true}\,.$$

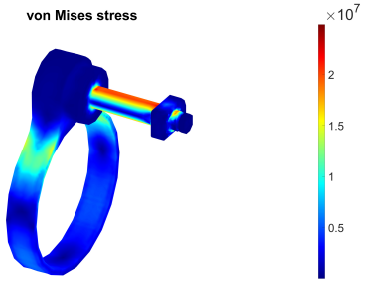
θ^* is a subset of optimal geometrical parameters, i.e. the parameters of the individuals that belong to the optimal Pareto front.

IV. Results

In this section, we apply the implemented framework to our use case, and the results are subjected to validation. We discuss the generation of the optimal Pareto fronts and their dependency on the geometric parameters. We compare the output parameters with the original design. The code to reproduce the results is available online [31]



(a) Stress distribution in the jetpack bracket.



(b) Stress distribution in the forearm support.

Fig. 4: FEM analysis for stress distribution in the generated CAD components.

A. Framework Specifications

The proposed framework is implemented using a Windows operating system equipped with an Intel(R) Core(TM)i7-10875H processor. We have modeled and parameterized CAD using CREO version 7. The scripts to evaluate stresses and generate the URDF models, as well as the flight controller and simulator, are implemented using Matlab and Simulink R2023b. The entire framework uses modeFrontier version 2020R3 to communicate with the individual nodes and generate the optimal population. The flight controller presented in Section II was taken from [29] and has been used to simulate the desired trajectory of $42\,\mathrm{s}$ as described in Section III.

From an initial random population of 25, we span 40 generations for a total of 1000 individuals. In the optimization process, 226 designs were deemed unfeasible due to the safety factor and QP failure constraits. The overall simulation time was 60 hours. The optimal Pareto front yields the last 75 individuals optimized for the defined objectives.

B. Co-Design Output

The distribution of individuals across the three objectives defined by Eq. (6) in Section III is depicted in Fig. 6a. The plot shows the distribution of individuals with respect to velocity error and momentum error. The optimized individuals have been highlighted in color proportional to thrust objective.

To understand the impact of the optimized variables on the output objectives, a spider plot of the four selected Pareto optimal designs along with the original is generated and illustrated in Fig. 6b. An observation can be made, that in order to minimize the momentum and thrust objective, the input parameters need to be lower and on the contrary minimizing the velocity objectives demands the input variables to be higher. This can be observed clearly from the area of distribution of input and output parameters in individual designs depicted in spider plot {OPTIM1, OPTIM2, OPTIM3, OPTIM4} are selected optimal designs that shall be used to validate the optimization against the original design. The corresponding optimized parameter values can be seen in Table II.

TABLE II: Table of designs selected to be validated and their optimized geometrical parameters.

Design	Angle [°]	Distance [mm]	Offset [mm]	Length [mm]
Original	15	42	80	108
OPTIM1	1	47	88	50
OPTIM2	2	40	94	50
OPTIM3	1	48	100	130
OPTIM4	8	96	100	146

C. Validation

To validate the performance of the optimized designs, we test their flight capability with five different trajectories. These trajectories, outlined below, aim to assess the robot's capabilities in both linear and angular maneuvers along with the possibility to quickly change directions.

- *Trajectory-1*: takeoff → move forward → move down → move backward → move down;
- *Trajectory-2*: takeoff while rotating clockwise along the yaw axis → move forward → rotate the yaw anticlockwise → move backward → move down while rotating the yaw anticlockwise.

Trajectory-1 is designed to assess the robot's ability to smoothly track a continuous linear path. In contrast, *Trajectory-2* is more demanding, requiring the robot to navigate both linear and angular motions simultaneously. *Trajectory 3 to 5* are generated using the same motion patterns of *Trajectory 1 and 2*, but with different orders and different settling times to complete each action. To ensure consistency in the measurement of the objectives, we modeled all the flight envelopes to have a total simulation time of $42\,\mathrm{s}$.

Results of the validation procedure can be seen in Fig. 7. The plots in Figs. 7a - 7c show that the four optimal designs have, in average, lower momentum error and overall thrust than the original design. Concerning the velocity error, the plot in Fig. 7b demonstrates that OPTIM1 and OPTIM2 have marginally higher errors and OPTIM3 and OPTIM4 are marginally lower than the original design.

Therefore, through the optimization, we were able to obtain designs that have better performance for what concerns momentum tracking, which is directly related to flight performances, and sum of thrust, which connects instead with the energy spent by the robot for flight.

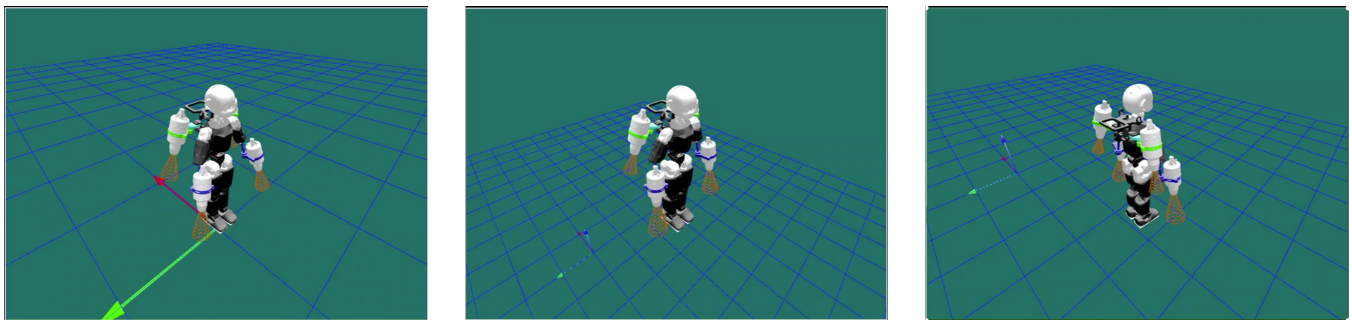
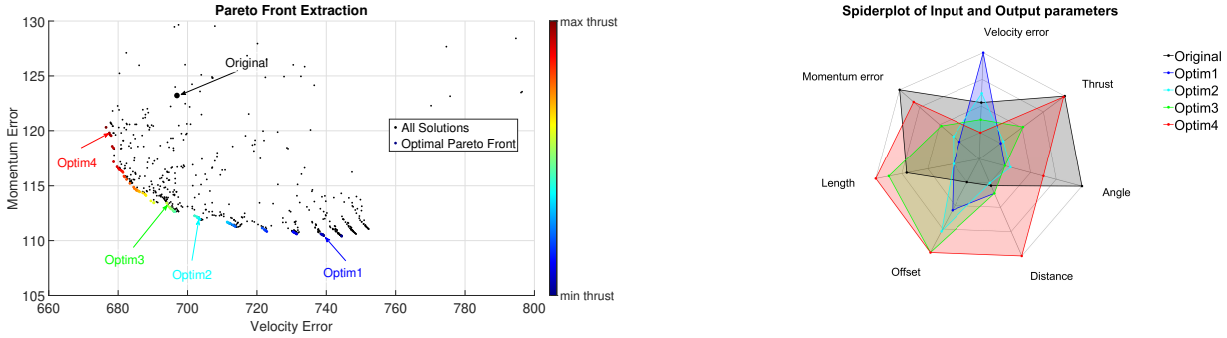
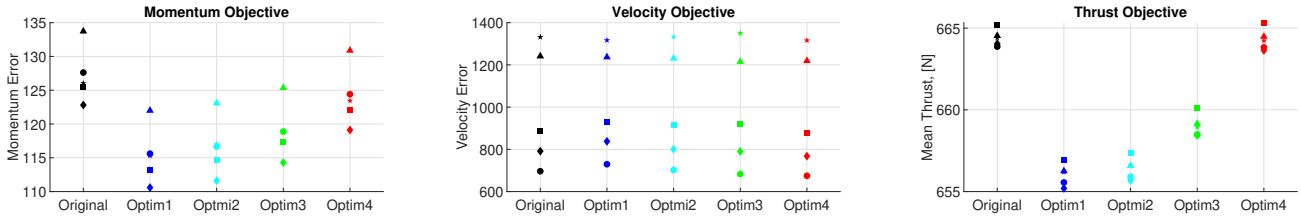


Fig. 5: Snapshots of the flight simulator and the flight envelope used for design optimization.



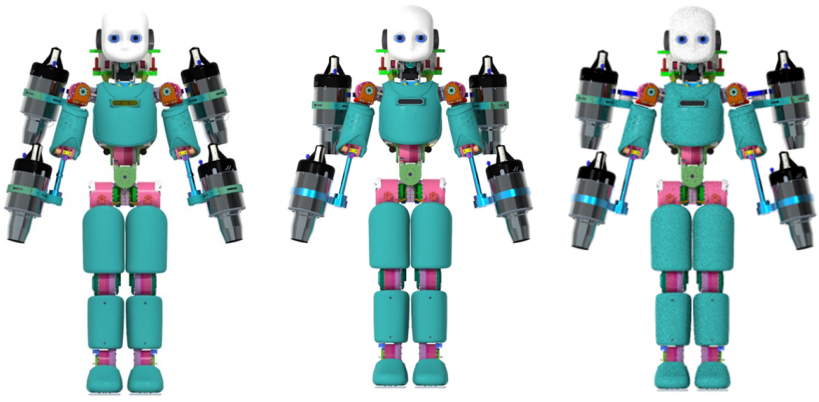
(a) (b)

Fig. 6: (a) Result of the co-design framework: each point represents a feasible individual. The colored individuals belong to the optimal Pareto front. The colors are selected proportionally to the thrust, The original robot design, reported for comparison, is dominated by the optimal Pareto front individuals. (b) Spider plot showcasing the original design and four optimal designs.



(a) (b) (c)

Fig. 7: Plot of four optimal designs compared with the Original over •*Trajectory-1*, ■*Trajectory-2*, ⋆*Trajectory-3*, ⧫*Trajectory-4*, and ▲*Trajectory-5*. (a) Momentum objective, (b) Velocity objective, (c) Thrust objective.



(a) Original (b) Optim3 (c) Optim4

Fig. 8: Updated CAD with Optimal designs.

D. CAD Output

Since the entire framework uses parameterized CAD, updating the CAD model of the robot and prototype the optimized pieces does not require, in principle, any additional adjustments. STEP and STL formats also allow the propagation of geometry into many modeling software and proceed directly toward manufacturing. In figs. 8 we have the show the updated CAD of the two optimal designs OPTIM3 and OPTIM4 against the Original.

V. CONCLUSIONS

We have proposed a co-design framework aimed at optimizing CAD parts of a jet-powered humanoid robot. Leveraging the NSGA-II algorithm within modeFRONTIER, our optimization focuses on enhancing the flight capabilities of the robot through parametrized CAD geometries.

Validation using various flight envelopes has demonstrated improved performances of the optimized CAD models compared to the baseline CAD. However, a current limitation of our framework is its inability to optimize the entire robot assembly. In future work, we aim to explore methods for extending the optimization procedure to encompass more robot parts without significantly increasing complexity and computational time. Additionally, we are interested in further integrating CAD-related features and analysis into the optimization process.

REFERENCES

[1] A. Gupta, S. Savarese, S. Ganguli, and L. Fei-Fei, "Embodied intelligence via learning and evolution," *Nature communications*, vol. 12, no. 1, p. 5721, 2021.

[2] M. Sitti, "Physical intelligence as a new paradigm," *Extreme Mechanics Letters*, vol. 46, p. 101340, 2021.

[3] J. T. Allison and D. R. Herber, "Special section on multidisciplinary design optimization: multidisciplinary design optimization of dynamic engineering systems," *AIAA journal*, vol. 52, no. 4, pp. 691–710, 2014.

[4] M. Garcia-Sanz, "Control co-design: an engineering game changer," *Advanced Control for Applications: Engineering and Industrial Systems*, vol. 1, no. 1, p. e18, 2019.

[5] T. Chen, Z. He, and M. Ciocarlie, "Co-designing hardware and control for robot hands," *Science Robotics*, vol. 6, no. 54, p. eabg2133, 2021.

[6] ROS Wiki, "URDF Documentation." `https://wiki.ros.org/urdf`.

[7] L. Kunze, T. Roehm, and M. Beetz, "Towards semantic robot description languages," in *2011 IEEE International Conference on Robotics and Automation*, pp. 5589–5595, IEEE, 2011.

[8] A. Spielberg, B. Araki, C. Sung, R. Tedrake, and D. Rus, "Functional co-optimization of articulated robots," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5035–5042, IEEE, 2017.

[9] C. Sartore, L. Rapetti, F. Bergonti, S. Dafarra, S. Traversaro, and D. Pucci, "Codesign of humanoid robots for ergonomic collaboration with multiple humans via genetic algorithms and nonlinear optimization," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pp. 1–8, IEEE, 2023.

[10] G. Fadini, T. Flayols, A. Del Prete, N. Mansard, and P. Souères, "Computational design of energy-efficient legged robots: Optimizing for size and actuators," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9898–9904, IEEE, 2021.

[11] G. Fadini, T. Flayols, A. Del Prete, and P. Souères, "Simulation aided co-design for robust robot optimization," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11306–11313, 2022.

[12] G. Saurel, J. Carpentier, N. Mansard, and J.-P. Laumond, "A simulation framework for simultaneous design and control of passivity based walkers," in *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, pp. 104–110, IEEE, 2016.

[13] S. Chand and D. Howard, "Multi-level evolution for robotic design," *Frontiers in Robotics and AI*, p. 192, 2021.

[14] A. Zhao, J. Xu, M. Konaković-Luković, J. Hughes, A. Spielberg, D. Rus, and W. Matusik, "Robogrammar: graph grammar for terrain-optimized robot design," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–16, 2020.

[15] J. Xu, A. Spielberg, A. Zhao, D. Rus, and W. Matusik, "Multi-objective graph heuristic search for terrestrial robot design," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9863–9869, IEEE, 2021.

[16] F. Bergonti, G. Nava, V. Wüest, A. Paolino, G. L'Erario, D. Pucci, and D. Floreano, "Co-design optimisation of morphing topology and control of winged drones," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8679–8685, IEEE, 2024.

[17] G. Fadini, S. Kumar, R. Kumar, T. Flayols, A. Del Prete, J. Carpentier, and P. Souères, "Co-designing versatile quadruped robots for dynamic and energy-efficient motions," *https://laas.hal.science/hal-04162737*, 2023.

[18] C. Sartore, L. Rapetti, and D. Pucci, "Optimization of humanoid robot designs for human-robot ergonomic payload lifting," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pp. 722–729, IEEE, 2022.

[19] J. Xu, T. Chen, L. Zlokapa, M. Foshey, W. Matusik, S. Sueda, and P. Agrawal, "An end-to-end differentiable framework for contact-aware robot design," *arXiv preprint arXiv:2107.07501*, 2021.

[20] A. Sathuluri, A. V. Sureshbabu, and M. Zimmermann, "Robust co-design of robots via cascaded optimisation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11280–11286, IEEE, 2023.

[21] M. O. Gulec and S. Ertugrul, "Pareto front generation for integrated drive-train and structural optimisation of a robot manipulator conceptual design via nsga-ii," *Advances in Mechanical Engineering*, vol. 15, no. 3, p. 16878132231163051, 2023.

[22] R. Saravanan, S. Ramabalan, N. G. R. Ebenezer, and C. Dharmaraja, "Evolutionary multi criteria design optimization of robot grippers," *Applied Soft Computing*, vol. 9, no. 1, pp. 159–172, 2009.

[23] Y. Sun, C. Zong, F. Pancheri, T. Chen, and T. C. Lueth, "Design of topology optimized compliant legs for bio-inspired quadruped robots," *Scientific Reports*, vol. 13, no. 1, p. 4875, 2023.

[24] D. Pucci, S. Traversaro, and F. Nori, "Momentum control of an underactuated flying humanoid robot," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 195–202, 2017.

[25] H. A. O. Mohamed, G. Nava, G. L'Erario, S. Traversaro, F. Bergonti, L. Fiorio, P. R. Vanteddu, F. Braghin, and D. Pucci, "Momentum-based extended kalman filter for thrust estimation on flying multibody robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 526–533, 2021.

[26] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," in *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18–20, 2000 Proceedings 6*, pp. 849–858, Springer, 2000.

[27] G. Nava, L. Fiorio, S. Traversaro, and D. Pucci, "Position and attitude control of an underactuated flying humanoid robot," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pp. 1–9, IEEE, 2018.

[28] T. Hui, A. Paolino, G. Nava, G. L'Erario, F. Di Natale, F. Bergonti, F. Braghin, and D. Pucci, "Centroidal aerodynamic modeling and control of flying multibody robots," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2017–2023, IEEE, 2022.

[29] Artificial and Mechanical Intelligence Laboratory (IIT), "ironcub-mk1-software." `https://github.com/ami-iit/ironcub-mk1-software`.

[30] JetCat, "P250-Pro-S." `www.jetcat.de/en/productdetails/produkte/jetcat/produkte/Professionell/p250%20pro%20s`.

[31] P. R. Vanteddu, G. Nava, F. Bergonti, G. L'Erario, A. Paolino, and D. Pucci, "From cad to urdf: Co-design of a jet-powered humanoid robot including cad geometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024. `https://github.com/ami-iit/paper_vanteddu_2024_iros_cogenerative_cad`.