Data-Driven Design of a Prosthetic Finger Joint

Matthieu Gachet, Juliette Hars, Olivia Zheng, Marc Domenjoz
École Polytechnique Fédérale de Lausanne

I. Introduction

THE HUMAN hand is a remarkably powerful and complex limb. Together, the hands contain 26% of the body bones and 10% of its muscles. Throughout human history, they have enabled humanity to evolve, and today, they are crucial for maintaining a high quality of life.

Therefore, the development of hand prosthesis technology is a significant field of research. As explained by [1], since the natural hand weighs only around 400 grams, maintaining a light weight should be a priority in designing a bionic hand. The same paper highlights that most finger mechanisms are comprised of four-bar linkages and, for the sake of simplicity and weight reduction, are typically actuated by a single motor. This leads researchers to design either under-actuated mechanisms or one-degree-of-freedom (DOF) fingers. Unfortunately, these types of mechanisms with one DOF are not shape-adaptive.

Compliant printed mechanisms appear to have the advantage of being shape-adaptive while requiring only a single motor for actuation. Moreover, as noted by [2], these mechanisms enable the development of affordable hand prostheses, thereby improving living conditions for a larger number of people.

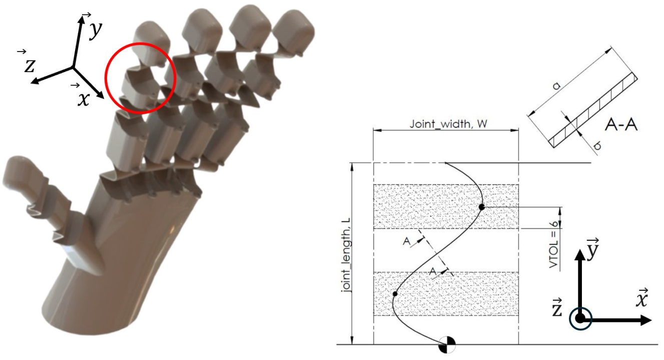
Different paradigms are possible for compliant 3D-printed orthoses. [3] developed a solution that implements inflatable fingers, resulting in a highly shape-adaptive design. On the other hand, [4] created a more conventional mechanism using bars and flexible hinges. In all cases, compliant fingers allow for passive finger extension due to their elasticity, therefore simplifying the actuation.

In this paper, curved flexible hinges will be employed to connect the segments of the fingers, as illustrated in Figure 1a. To optimize weight, the flexion stiffness, $k_{\theta,z}$, will be minimised, enabling the use of small motors. Simultaneously, the stiffness in abduction/adduction direction, $k_{\theta,x}$, will be maximized, as it will not be controlled. Additionally, a constraint was introduced to minimize parasitic movements of the developed flexible pivot compared to an ideal one, $RMSE$ (see Eq. 1).

Due to the multitude of objectives and the complexity of the design space, the problem is addressed using data-driven methods.

II. Design Problem

As depicted by Figure 1a, the developed finger is constituted of 3 phalanxes linked with ”S” shaped flexible blades. This work only focuses on the optimisation of one joint. As shown in Figure 1b, the joint is encoded through a rectangle of dimension a and b extruded along a spline defined by N control points.



(a) Schematic drawing of the complete prosthesis.

(b) Representation of the encoding of a segment.

A. Encoding

The joint is encoded as a chromosome, as shown in Table II. The design variables and parameters are denoted in Table I. In particular, a minimum vertical distance between each pair of points, $VTOL$, was set to ensure feasability of the joint. The intermediary variable $l_{left} = VTOL * (N - i - 1), i \in [0, N-1] \in \mathbb{N}$ has been used to define the y_i bounds.

Fixed parameter	**Value**
Finger width W	20 [mm]
Spline height a	20 [mm]
Min distance b/w points $VTOL$	6 [mm]
Initial and final control point pos. $\begin{pmatrix} x_0 \\ y_0 \end{pmatrix}, \begin{pmatrix} x_N \\ y_N \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ L \end{pmatrix}$
Optimisation variable	**Bounds**
Number of control points N	$[2, 6] \in \mathbb{N}$
Control point position $\begin{pmatrix} x_i \\ y_i \end{pmatrix}$	$\begin{pmatrix} [-W/2, W/2] \\ y_{i-1} + VTOL, L - l_{left} \end{pmatrix}$
Joint length L	$[15[\text{mm}], 30[\text{mm}]] \in \mathbb{R}$
Spline width b[mm]	$0.8 + n * 0.4, n \in [0, 6] \in \mathbb{N}$

TABLE I: Design parameters and variables.

L	b	x_1	y_1	...	x_{N-1}	y_{N-1}

TABLE II: Chromosome representation of the joint.

The base fitness function is defined as in Equation 1, although tuned for the GA optimization step as described subsequently.

$$\text{f} = k_{\theta,z}^2 / k_{\theta,x} \cdot RMSE. \tag{1}$$

B. *Dimensionality of the design space*

The dimensionality of the design space depends on the number of spline points, and is equal to $2N - 2$. One dimension comes from the length of the spline L, another one from its thickness b. Additionally, there are N two-dimensional spline points, of which the first and last ones are fixed to constant values. The dimensionality is therefore computed as: $\mathbf{Dim} = 1{+}1{+}2{\cdot}(N{-}2) = 2N{-}2$. In a tentative dimensionality reduction, PCA was applied on the data, standardized before-hand, for N ranging from 2 to 6. As the dimensionality is dependent on N, PCA had to be applied for each N separately.

Results for N=4 are presented in Figure 2, showing that an a priori dimensionality of 6 may be reduced to 4 when keeping a cumulative explained variance of 95% at least. As N gets smaller, the dimensionality reduction becomes less and less relevant. For simplicity and time reasons, it was finally decided not to implement PCA, as it would have required running a data-driven optimization algorithm (GA in this case) for each N separately.

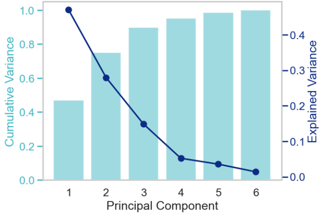


Fig. 2: Explained variance, $N = 4$

III. Data-driven plan and methods

The data-driven design of the prosthetic finger joint was divided in two steps, namely a Genetic Algorithm based optimization in parallel with an FEM simulation of the joint, followed by a Bayesian optimization algorithm with real-world testing in order to validate and improve on the results.

GA. Firstly, a genetic algorithm was implemented because of the large design space with both discrete and continuous parameters, potentially highly dimensional depending on the number of spline points, and that may have many local optima. A python script that generated automatic simulation parameters was linked to the ABAQUS software in order to test the fitness of each individual, in the loop with the GA algorithm (this loop is described in more details in section IV-B). The simulation allowed to measure joint displacements in both **x** and **y** directions, as well as the (vertical) reaction force at the base of the joint. These measurements allowed to compute the following simplified fitness function: f $= RMSE \cdot k_{\theta,z}^2 \cdot L^*$.

As in Equation 1, the fitness function depends on the $RMSE$, the root mean square error between the (**x**,**y**) coordinates of the designed joint trajectory and those of the ”ideal trajectory” covered by a circle regressed on the real trajectory, as shown in Figure 3. $k_{\theta,z}$, the angular stiffness of the joint, is computed thanks to the force and position measurements, and with respect to the virtual center of rotation, in red (Fig. 3). This value is squared to emphasize its importance in the optimization process. Because the ABAQUS simulation is in 2D, Equation 1 has been simplified by swapping $k_{\theta,x}$ with the inverse of L^* (in orange, Fig. 3). $L*$ approximates the length of the joint, computed by summing the straight distances between spline points. This adaptation is justified by the fact that stiffness $k_{\theta,x}$ is approximated as being inversely proportional to L^*.

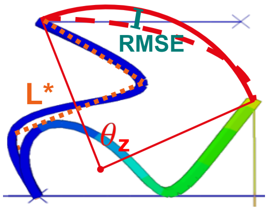


Fig. 3: Objective functions parameters

BO. Then, Bayesian optimization was implemented in parallel with real-world prototyping and testing, in order to validate the simulation/GA results and potentially improve on them. In fact, the two best results from GA were taken as starting point for the BO algorithm. BO was chosen for its ability to perform well under noisy and time-consuming evaluation of the objective. This evaluation was carried out by FDM printing of joint prototypes, and real-world testing with an appropriate test bench (Section 8a) used to measure the three objective terms as defined by Equation 1. Note that $k_{\theta,x}$ is here measurable, thus eliminating the need to approximate it with $1/L^*$.

IV. Simulation and GA

A. *Data capture - Simulation*

Pipeline. The data used in the scope of the Genetic Algorithm optimization was generated with an FEA ABAQUS simulation. Each joint was modelled as a 2D planar, deformable wire with PETG as an elastic material (with Young modulus E=1200 MPa and Poisson ratio ν=0.38 [5]). In order to perform a simulation as close as possible to the real life scenario, 2 rigid parts were added, namely a ”ceiling”, horizontal part placed at the top of the joint ($y = L$), and a ”floor”, similar but placed at the bottom of the joint ($y = 0$). The ceiling simulates the finger top part where the actuation cable passes, and the the floor simulates the finger base part. While the bottom of the joint was fixed with an ”encastre” boundary condition, a concentrated force with an incremental value from 0 to 50[N] was applied at the right extremity of the ceiling, to bend the joint and assess its displacement. Self contacts were used to prevent the joint from penetrating itself, and rigid contacts were applied to ensure no penetration of the floor. B22H meshing elements were used, and the vertical reaction forces and displacements were measured at each simulation step. Figure 4 summarizes the steps of the data collection and synchronization with the GA algorithm. The fitness of each individual is computed as presented in Section III.

Data quality. Finite element analysis with ABAQUS is based on assumptions and approximations, which may introduce disparities with real-world experiments. In this case, the simulation was made as close as possible to reality while keeping it simple, ie. 2D for example. Indeed, there is a clear tradeoff in performance between computation time and accuracy. By simplifying the model, although reducing the running time, one may approximate too much the reality and therefore obtain results that are far from it. Here, one simulation took between 15 to 30 seconds approximately, rather slow but acceptable in the case of the GA optimization. Additionally, other discrepancies may arise because of the material model, the mesh quality or the applied boundary conditions. In general however the trends in performance of the joints should be preserved with respect to reality, and should therefore allow for the optimization of the finger joint.

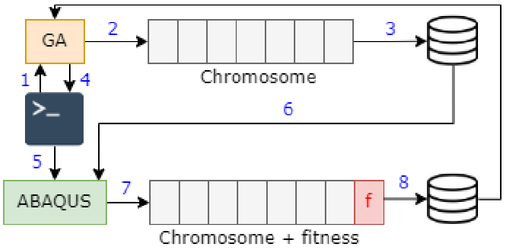


Fig. 4: Simulation and GA functional diagram. **1.** Terminal runs GA. **2.** GA creates one chromosome. **3.** The chromosome is saved to an external file. **4.** GA calls the terminal to run the ABAQUS simulation. **5.** ABAQUS is launched. **6.** ABAQUS reads the chromosome parameters from the external file. **7.** ABAQUS runs a simulation and computes the corresponding fitness. **8.** The chromosome and its fitness are saved to an external file. **9.** GA reads the external file and updates the population, then outputs a new chromosome to be tested.

B. GA method

The Genetic Algorithm was built following the steps demonstrated in Figure 5. Firstly, a random population of 50 individuals was created, with each chromosome containing the parameters presented in table II, with randomly assigned values following the corresponding constraints. After simulating individually each chromosome in ABAQUS and computing their fitnesses, GA is able to extract the "simulated" population, with a fitness assigned to each corresponding chromosome. It is important to note that to prevent any bias and to force the algorithm to choose joints with trajectories close to quarter circles, the fitness was automatically set to -1000 whenever the joint did not reach an angle of at least 70° after aplying the maximum force of 50[N]. From the population ordered by increasing fitness, a small number of individuals, corresponding to a ratio $r_{elit} \in \{0.05; 0.1\}$ of the whole population is copied to the new population. A ratio $r_{par} \in [0, 1 - r_{elit}]$ of the population is selected as parents and will undergo crossover and mutation operations with a certain probability, respectively $p_{cross} \in [0, 1]$ and $p_{mut} \in [0, 1]$.

Crossover. One-point crossover was implemented by randomly choosing a "cut index" along the length of the smallest chromosome. Then all entries before/on the left of the cut index were switched between the two parents to make the offspring. The chromosome encoding intently alternates x and y coordinates such as to make crossover possible. Indeed, only the constraint on the y_i coordinates (see table I) may be problematic. To palliate this issue, in case crossover led to an infeasible solution, the parent y coordinates were kept. Two-point crossover was also implemented in a similar manner, by randomly picking two cut indices along the length of the smallest parent.

Mutation. As mutation methods such as bit flip, swap, scrumble etc. could not be implemented in this case, the method simply consisted here of a random value mutation. Each gene in the chromosome is mutated with the probability p_{mut}. For L, a sample from the uniform random distribution is added to the current value as follows: $L' = L + N(0, \sigma)$ with $\sigma = 0.1 \cdot (L_{max} - L_{min})$. Point coordinates are mutated in a similar manner, as follows. $x'_i = x_i + N(0, 0.05 \cdot W)$, $y'_i = y_i + N(0, 0.1 \cdot (x_i + VTOL - (x_i - VTOL)))$ while ensuring that constraints are satisfied. The parameter b is mutated by randomly choosing another accepted discrete b value.

After applying crossover and mutation operators, the entire offspring is ready and added to the new population, which will be simulated, one individual after the other, and their fitness evaluated.

This process is repeated until reaching a number of generations of 30. Although this number could have been chosen higher, it allowed for convergence in all trials.

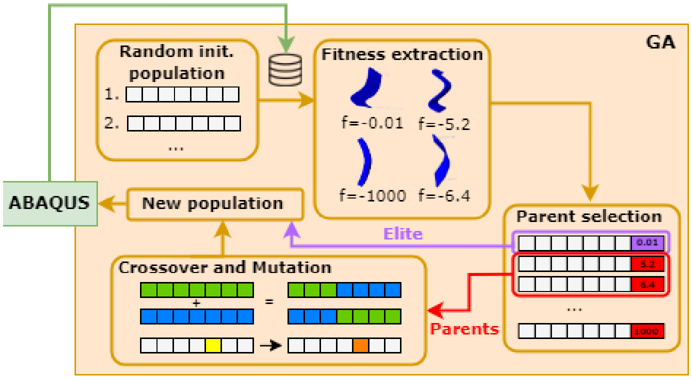


Fig. 5: Zoom in the GA algorithm primarily presented in figure 4. The different steps are presented, going from the creation of the population to the fitness extraction, parent selection, crossover and mutation.

C. GA results

Figure 6 shows the resulting mimimum fitness graphs of six GAs that were run with different parameters. A clear distinction can be made between the purple curves that correspond to a high crossover probability and the blue ones, that correspond to a low crossover probability. The green curve, the only one with two-point crossover with low probability ($p_{cross} = 0.05$), shows almost immediate convergence to a close to zero fitness. It may indicate that the best solutions are obtained by tuning the in-between points of the spline, and fixing L and b to optimal values. In general, a low crossover probability yielded faster convergence and lower fitness consistently across generations. Therefore, one may assume that keeping a low crossover probability while maintaining diversity with a high mutation one should improve the results. This assumption may be justified by the fact that crossover probably changes too drastically the individuals when just small modifications would be required to improve their fitness. Overall, it seems that a lot of solutions have almost equivalent fitnesses, indicating that the objective function may not have only one but several optima with almost identical fitness values.

To have some intuition on the quality of the results, two joints were ”hand designed” and their fitness assessed. The first design was intended to yield bad results, rather long and thicker than the minimum allowed with $L = 25$[mm] and $b = 1.2$[mm], and one spline point only at $x = 2, y = 11$ (without counting the fixed top and bottom points). This joint obtained a fitness after simulation of -28. The second design was intended to yield better results, with a shorter and thinner spline $L = 20$[mm], $b = 0.8$[mm], and two intermediate spline points at $x_1 = -5, y_1 = 5$, $x_2 = 5, y_2 = 5$. This joint obtained a fitness after simulation of -8.4, and is represented by a red dotted line on Figure 6. From this analysis, it can be noted that GA is able to improve on already ”good” results, by obtaining fitnesses close to 0.

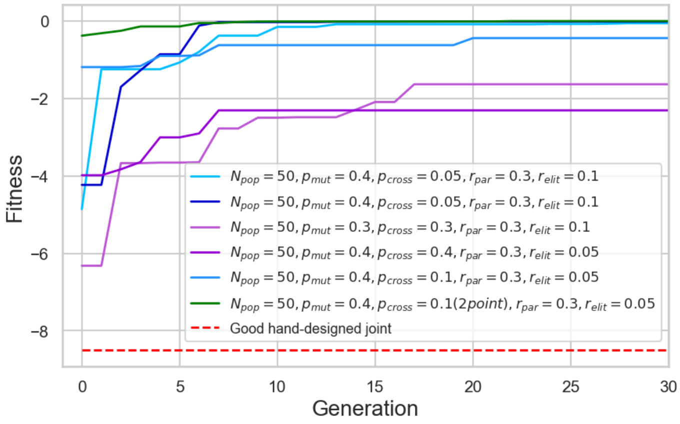


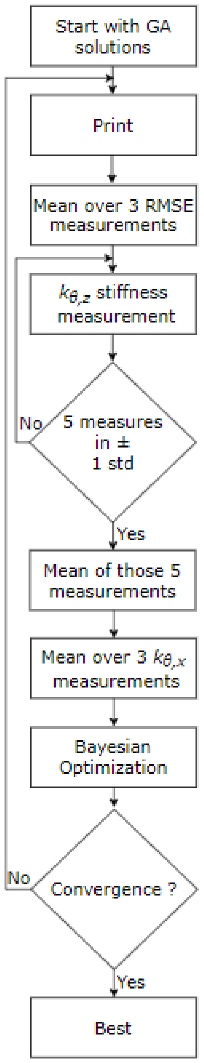
Fig. 6: GA minimum fitness graphs for different crossover and mutation probabilities, and different elite ratios.

V. Physical world optimization, BO

A. Data capture

1) Pipeline: To ensure systematic testing through iterations, a rigorous data acquisition pipeline was designed as depicted in Figure 7. Moreover, the utilization of a dedicated test bench (Fig. 8a) and graphical user interface (Fig. 8b) ensured repeatable measurements throughout the entire process.

This process starts with the two best designs from the Genetic Algorithm. After printing them in 3D using FDM technology and PETG, which takes 50 minutes, three RMSE measurements are performed and averaged. Actuation with an electric screwdriver ensures a constant speed, allowing a constant distribution of measured points by the camera along the entire trajectory. This is particularly crucial for $RMSE$, as measurements are taken at a fixed frame rate, and clustered points may significantly impact the results. For the same reason the $RMSE$ measurement is triggered automatically as soon as movement is detected and until the joint has travelled 1.15[rad].


Start with GA solutions
Print
Mean over 3 RMSE measurements
$k_{\theta,z}$ stiffness measurement
5 measures in ± 1 std
No
Yes
Mean of those 5 measurements
Mean over 3 $k_{\theta,x}$ measurements
Bayesian Optimization
Convergence ?
No
Yes
Best

[t]

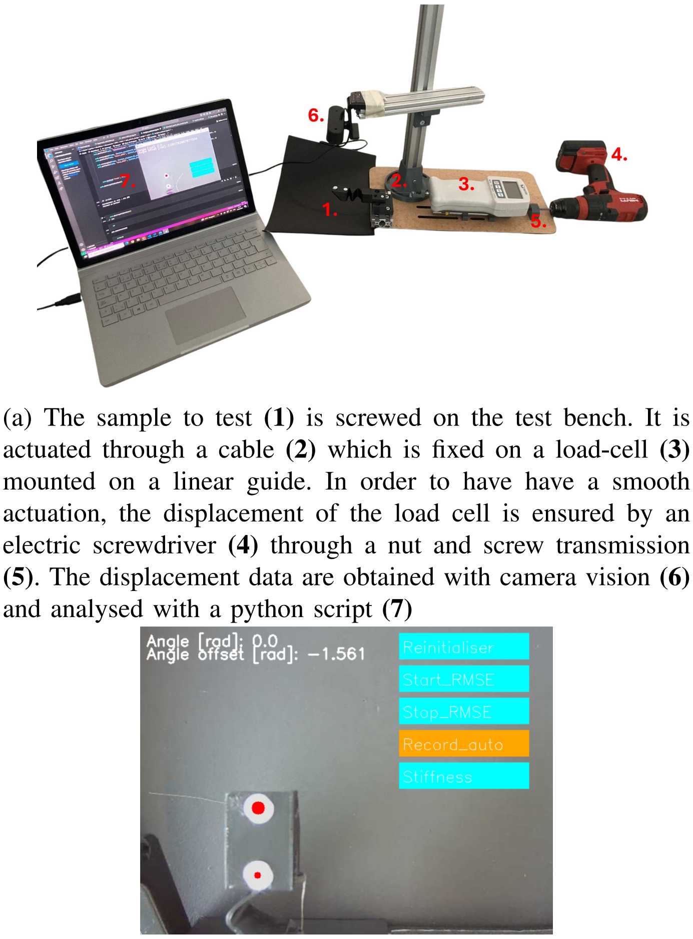
Fig. 7: Process followed for real life testing

Afterwards, the stiffness $k_{\theta,z}$ is determined by averaging the intermediate stiffness, $k_{\theta,z,\text{approx},i}$, measured along the force-deformation curve of the tested joint. (see Fig. 9.b)). This is repeated until five computed $k_{\theta,z}$ fall within one standard deviation of the others. These are then averaged. This approach helps eliminate outliers. Indeed, the force gauge used has a resolution of 0.5[N], which is not ideal given the relatively small forces required to actuate the joints, sometimes as low as 4[N]. Therefore, special care must be taken. For the same reason, measurements are always taken on the falling edges of the force gauge readings (see Fig. 9.a)), ensuring as much consistency as possible.

Note that the standard deviation is influenced by outliers: the more significant the outliers, the higher the standard deviation. This variability relaxes or hardens the validation criteria, leading to less repeatable measurement conditions from one joint to another. A potentially better approach to explore as next step, is to use a fixed standard deviation as the validation criterion for all measurements across the whole optimisation.

Finally, since $k_{\theta,x}$ is stiffer and therefore less sensitive to measurement noise, it is evaluated by simply averaging three successive measurements.

After evaluating these three values, which takes approximately 40 minutes, the joint fitness can be calculated according to Section V-B. Subsequently, it is printed, and this process is repeated until convergence occurs. However, due to time constraints, the algorithm was stopped after 25 iterations, without reaching convergence. Potential explanations for this lack of convergence are further detailed in section VI.



(a) The sample to test **(1)** is screwed on the test bench. It is actuated through a cable **(2)** which is fixed on a load-cell **(3)** mounted on a linear guide. In order to have have a smooth actuation, the displacement of the load cell is ensured by an electric screwdriver **(4)** through a nut and screw transmission **(5)**. The displacement data are obtained with camera vision **(6)** and analysed with a python script **(7)**

(b) This interface calculates RMS error and stiffness by enabling users to intuitively input force values at various joint positions.

Fig. 8: Data acquisition setup

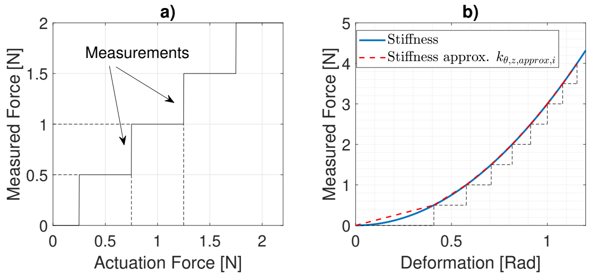


Fig. 9: **a)** measurement on falling edge. **b)** $k_{\theta,z}$ stiffness measurement points.

2) Data quality: As explained above, each objective function component is obtained by averaging at least three measurements (e.g. $RMSE_{iter}$ = mean($RMSE_{iter,m}$) m=1,2,3). Therefore, we can compute the standard deviation of these measurements (e.g. std($RMSE_{iter,m}$) m=1,2,3) and compare it with the average. The greater the difference in terms of orders of magnitude, the more reliable the measurements are. Table III compares the mean and the std dev of all three components by averaging them across all joints. It is noteworthy that there are less than 2 orders of magnitude between the computed value and the std dev, highlighting the low precision on the values used in BO optimisation.

	$RMSE$	$k_{\theta,z}$	$k_{\theta,x}$
Overall mean	0.00141	5.005	38.824
Overall Std	0.0001	0.2878	1.262
Std/mean	7.01%	5.7 %	3.25%

TABLE III: Caption

The resolution of the camera (640x480 pixels) and force gauge (0.5[N]) are the first factors explaining the lack of measurement precision.

From a more qualitative point of view, the fact that different printers were used throughout the process, as well as variations in the PETG filament brand, may have an impact on physical properties that were not quantified.

Lastly, during the measurement of $k_{\theta,x}$, the joint sometimes twisted on itself due to its ”S” shape. This parasitic movement was not accounted for in the measurements.

B. BO method

The Bayesian optimization algorithm uses the same encoding as that of the GA (refer to Table II). However, as mentioned in Section III, the objective function now utilizes measured values of $k_{\theta,x}$ as defined by Equation 1.

The surrogate model used is the Gaussian process model from the Python scikit-learn library, with a Radial Basis Function (RBF) as the kernel. Both the model and kernel were chosen as they are popular options for extracting complex patterns and relationships to uncertainty from datasets. For the acquisition function, the expected improvement function was utilized as it takes into consideration both probability of improvement as well as how large that improvement will be.

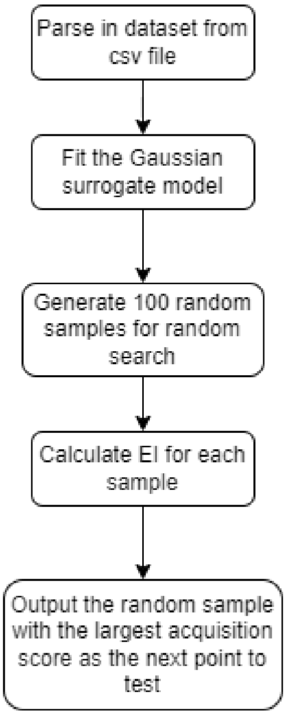

Parse in dataset from csv file
Fit the Gaussian surrogate model
Generate 100 random samples for random search
Calculate EI for each sample
Output the random sample with the largest acquisition score as the next point to test

Fig. 10: Flowchart of the Bayesian optimization algorithm

The Bayesian algorithm, as shown in Fig. 10, begins by parsing in the existing dataset from an external file. This dataset contains the parameters for each test, along with the measured $k_{\theta,z}$, $k_{\theta,x}$, and $RMSE$ values, and finally the objective function value. The existing dataset is used to fit the Gaussian process surrogate model. Then, 100 random potential samples are generated according to our boundaries and constraints. The expected improvement is calculated for each sample and the one with the best score is selected as output to be tested next. Once the test is physically conducted, the dataset is updated with the new results and the algorithm can be used again.

C. BO results

The evolution of BO can be seen in Fig. 11. Fitness values above 0.001 have been set to 0.001 to give a better view of the evolution. The two best solutions are circled in red, with their corresponding designs pictured beside the points. It is visible from this graph that there is no convergence, and there are multiple local minima with little difference between them.

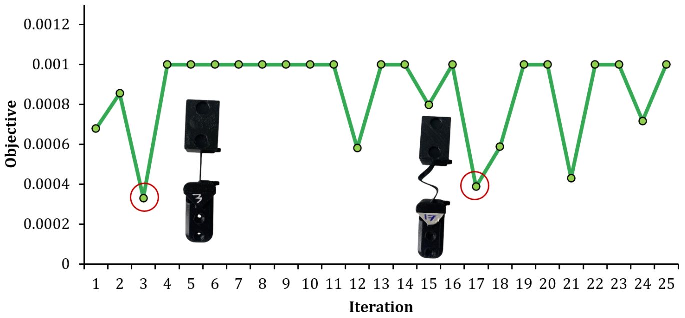


Fig. 11: Evolution of the fitness

VI. Discussion and Conclusion

It was observed that Bayesian optimization found a better solution than the solution found with genetic algorithms, despite the Bayesian optimization having not converged in the end.

With the Bayesian optimization, the measurement noise from the tests likely increased the difficulty of finding a global minimum. The high dimensionality of the parameter space may be another reason for the lack of convergence, as BO tends to perform better on datasets with a smaller number of dimensions. There may also have been limited room for improvements with the BO as it started by using the two best solutions from the GA. Additionally, the rigidity approximation done for the sake of 2D simulation for the GA may also have had an impact, especially on the difference between the solutions output by the GA and the BO.

This study suggests that a more effective approach would involve using genetic algorithms not only to identify initial joints for physical testing but also to fix certain design variables before proceeding with Bayesian optimization. This method would enable rapid reduction of the design space dimensionality, thus streamlining convergence for Bayesian optimization.

A. Possible Improvements and Next Steps

This work opens up a whole field of ideas that could improve the final result and take it even further.

Firstly, as mentioned previously, the accuracy of the ABAQUS simulation depends highly on its complexity as well as on the chosen parameters. In completion to this work, a study on the accuracy of the simulation to quantify the sim-to-real gap may be performed, in order to gain insights on the quality and relevance of the data, and potentially improve GA results.

Additionally, the crossover and mutation operations applied in GA might be improved by incorporating constraint knowledge directly in order to extend the explored space. Indeed, eliminating infeasible candidates is inefficient and might cause the algorithm to stagnate in local minima. Therefore, feasibility could be preserved for example through custom operators designed specifically to enforce the constraints, or through repair mechanisms if an efficient and reproducible method were to be found. A study could also be made on the impact of constraint relaxation, in order to assess whether it would unlock some precendently unexplored potentially better solutions.

In the physical world optimization, to improve the current set-up, integrating a more accurate load cell connected to a computer, along with a motor to actuate the joint, would significantly improve the automation of the process, resulting in a more robust and faster measurement setup.

Then, from a purely mechanical point of view, it is important to note that the current joint, due to its shape, has more than just the $\theta_{\mathbf{z}}$ degree of freedom. Despite maximizing its stiffness, $\theta_{\mathbf{x}}$ remains a degree of freedom, along with the three translations, even if $\mathbf{x}$ and $\mathbf{z}$ are highly stiff. It would be prudent to carefully assess the need for stiffening them in future. Noting that the current flexibility in $\theta_{\mathbf{x}}$ may be utilized for future implementation of abduction/adduction of the fingers.

Finally, exploring different manufacturing solutions may lead to improved designs. This includes studying the use of various materials and printing methods. Further reflections could be conducted on replacing compliant parts with part laser cut or stamped from spring steel sheet, which could result in more robust fingers, albeit at the cost of creating a multi-part finger.

References

[1] S. R. Kashef, S. Amini, and A. Akbarzadeh, “Robotic hand: A review on linkage-driven finger mechanisms of prosthetic hands and evaluation of the performance criteria,” *Mechanism and Machine Theory*, vol. 145, p. 103677, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X19322839

[2] J. ten Kate, G. Smit, and P. Breedveld, “3d-printed upper limb prostheses: a review,” *Disability and Rehabilitation: Assistive Technology*, vol. 12, no. 3, pp. 300–314, 2017. [Online]. Available: https://doi.org/10.1080/17483107.2016.1253117

[3] Instructables.com, “3d print an artificial muscle robot hand,” 2024, retrieved June 14, 2024. [Online]. Available: https://www.instructables.com/3d-Print-An-Artificial-Muscle-Robot-Hand/

[4] M. Groenewegen, “Design of a compliant, multi-phalanx underactuated prosthetic finger,” Master’s thesis, Delft University of Technology, May 22 2014, embargo date: 2014-06-05. [Online]. Available: http://resolver.tudelft.nl/uuid:384942fc-467c-4bba-b052-3eef484a4b5c

[5] J. Mercado-Colmenero, M. La Rubia, E. Mata-Garcia, M. Rodriguez-Santiago, and C. Martin-Doñate, “Experimental and numerical analysis for the mechanical characterization of petg polymers manufactured with fdm technology under pure uniaxial compression stress states for architectural applications,” *Polymers (Basel)*, vol. 12, no. 10, p. 2202, Sep 2020.