Multi-objective design optimisation of a delta coordinate measurement machine

Jakobus Murray Louw*[1], *Willis* de Ronde[1], *Stephen* Marais[1], and *Thabisa* Maweni[1]

[1]Council for Scientific and Industrial Research, Centre for Robotics and Future Production, Pretoria, South Africa

Abstract. A systematic approach for multi-objective optimisation in machine design is presented and further demonstrated through a case study on a delta Coordinate Measurement Machine (CMM). Employing the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [1], the methodology aims to balance competing objectives like measurement accuracy, motion resolution, and machine size. Through an iterative process and simulation-guided parameter refinement, new Pareto optimal solutions are identified at concurrent decision-making steps to reach a final design solution. The results from four concurrent simulations are presented, where each simulation is used to reduce the input range of a specific design variable. The visualizations reveal complex relationships between the design variables and outlier clusters are identified and excluded from the solution space. Furthermore, the results demonstrate how the solution set is systematically reduced to reach a Pareto optimal design. Overall, the proposed process offers a structured framework for addressing the complexities of multi-objective machine design, as evidenced by its successful application in optimising a delta CMM.

1 Introduction

Multi-Objective Optimisation (MOO) involves simultaneously optimising multiple and often conflicting objectives to find a set of solutions that represent trade-offs between these objectives. The theory of MOO initially emerged in mathematical economics and has since been applied to problems in various other fields. Solving complex machine design problems has been a particularly popular application of MOO [2-5]. However, many projects use unique approaches that may not be reusable for new design tasks. This paper aims to compose a generalised and systematic process for multi-objective optimisation of machine design tasks. This methodology will also be demonstrated on a specific delta parallel mechanism Coordinate Measurement Machine (CMM) design.

* Corresponding author: mlouw2@csir.co.za

1.1 Optimisation algorithms

Optimisation algorithms can be broadly categorised into gradient-based and gradient-free approaches. Gradient-based methods leverage the derivative information of mathematical objective functions to guide the search for optimal solutions. Gradient-free approaches, like particle swarm optimisation, Genetic Algorithms (GA) or other evolutionary algorithms, are particularly effective for optimising non-linear, and discontinuous models due to their ability to efficiently search multi-dimensional solution spaces without requiring explicit mathematical expressions [5]. As an example, a previous study [6] demonstrated the application of a GA with an object-orientated model of a Broadband Powerline Communication network to optimize the placement of repeaters. Furthermore, in review of MOO algorithms used in mechanical engineering problems [2], evolutionary algorithms, and more specifically NGSA-II [1], were the most used across these problems. According to the authors, NSGA-II appeared in 32 out of 90 research studies detailed [2].

1.2 MOO applied to machine design

The MOO approach has been useful for engineering problems where multiple conflicting objectives must be balanced in the design. In [5], MOO was used to select optimal components for efficient configuration as part of a multi-copter design process. The authors implemented MOO by using a GA to optimise multiple design objectives, including flight time, power consumption and price.

In a different study [3], the content of rare-earth material was successfully reduced in an optimised permanent magnet-assisted machine design, while maintaining the same average torque and minimising demagnetisation risk at low operating temperature constraints. In this case, a differential evolution algorithm was used to generate candidate designs.

1.3 Delta CMM

In manufacturing industries, CMMs are used for the three-dimensional inspection of physical components. These are mechanical systems which facilitate the movement of a measuring probe to ascertain the coordinates of points on the surface of a workpiece with high accuracy and precision. Fig. 1 shows an example of a large commercial cartesian CMM.

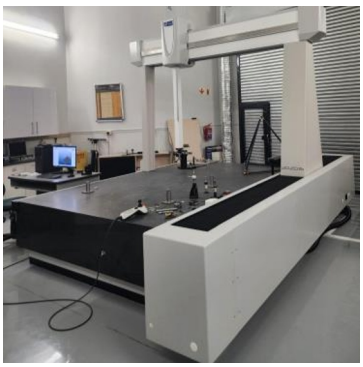

Fig. 1. Example of a cartesian CMM.

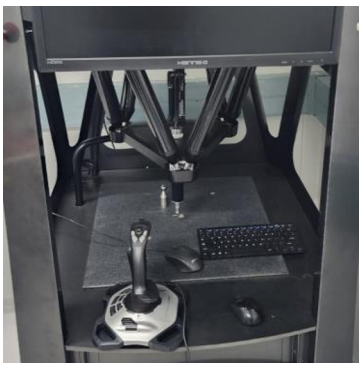

Fig. 2. Example of a parallel mechanism CMM.

Increased demands in the development of complex and miniaturised mechanical components have opened a gap for measuring technologies that can meet the need for high precision and accuracy on miniaturised parts [7].

Delta CMMs use a parallel robot mechanism for the benefits of high stiffness and low moving inertia. Fig. 2 shows an example of a commercial CMM using a parallel robot mechanism. The delta CMM design considered in this paper has its main structure built using parallelogram mechanisms and a moving platform that has three translational degrees of freedom.

2 Methodology

This section outlines the proposed methodology for optimising the design of a machine. Fig. 3 illustrates this process as a flow diagram. Initially the simulation model is defined, and the optimisation problem is formulated. Thereafter, the optimisation algorithm is executed, and its results are visualised. Based on these results, design decisions are made to refine the solution space by either fixing decision variable values or by adding constraints. Thereafter, the optimisation problem is adjusted accordingly and rerun until the solution space has been reduced enough for a single solution to be selected. Finally, a sensitivity analysis is applied to ensure that the chosen solution is robust and reliable, assessing how variations in input parameters affect the performance and feasibility of the optimized design. If the remaining solution space is too unstable, some design decisions need to be relaxed. This is required to broaden the solution space and find a solution that is robust.

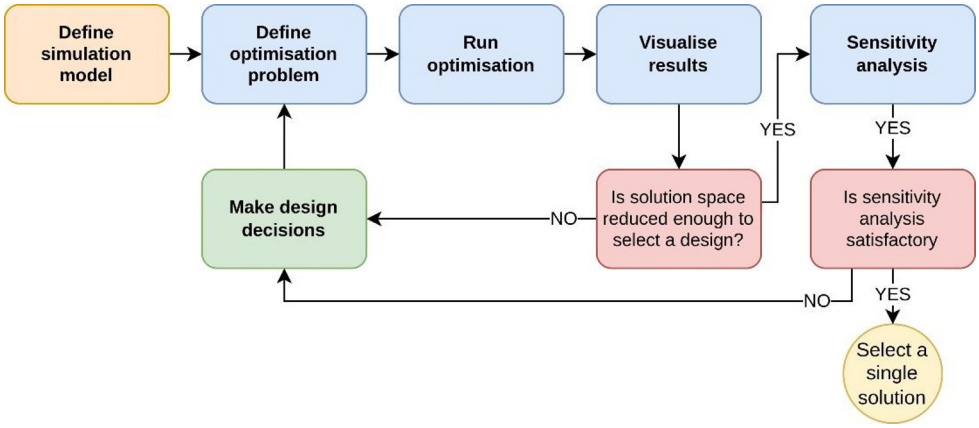

Fig. 3. Overview of the proposed methodology.

2.1 Simulation model creation

A machine simulation model must be created to initiate the design optimisation process. Simple machines can be described by closed-form mathematical equation models, which analytically express the relationships between design variables, objectives, and constraints. These models lend themselves well to gradient-based optimisation methods [8]. However, many real-world machines are not easily modelled by closed-form mathematical equations.

Object-oriented programming (OOP) models provide a flexible approach for modelling complex, real-world machines by describing components and interactions as classes and functions. Algorithms like genetic algorithms or particle swarm optimisation are commonly used with OOP models. In this study, an OOP class structure used to define an inverse kinematic model of the CMM using vector mechanics [9]. Furthermore, a constructor is implemented to instantiate build parameters, and functions are implemented to calculate metrics for objectives and constraints.

2.1.1 Class structure creation

An OOP class structure is developed to represent the machine, with each machine subsystem represented as nested classes within the main class structure. This facilitates hierarchical organization and encapsulation of system components. Within the main class constructor, design parameters are instantiated as parametric class variables, which are passed down to member objects to define the parameters of each part of the machine.

Python [10] was chosen as the primary programming language for this study due to its low complexity and its ease of use, making it accessible to researchers with varying levels of programming expertise. Additionally, Python contains a vast array of optimisation and numerical libraries, such as NumPy [11] and Pymoo [12], which provide robust tools for numerical modelling and multi-objective optimisation.

Fig. 4 shows a schematic with labelled parameters of the delta CMM considered in this study. The machine consists of three pillars that rigidly connect the base plate to the roof plate. Each pillar contains a linear actuator that accurately moves up and down the pillars. The end-effector is connected to the three pillars with three parallel mechanism arms, such that the relative vertical motion of the linear actuators causes the 3D motion of the end-effector. A touch probe is attached to the bottom of the end-effector, which provides touch feedback when measuring objects in the work envelope.

For position measurements, three laser sensors are mounted to three sensor arms with horizontal and vertical components. These linear sensors measure the distance from their reference points to the end effector touch probe as it moves in the work envelope. The machine is designed with a cylindrical work envelope based on a minimum radius and height, such that the machine can move the tip of the touch probe anywhere within the work envelope. The work envelope is located on the base plate in the centre of the machine.

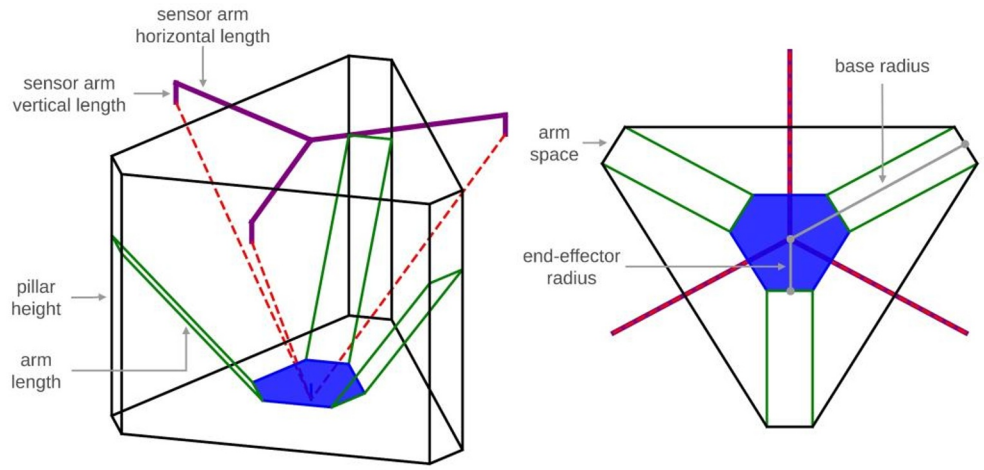

Fig. 4. Schematic of the delta CMM with labelled designed parameters.

2.1.2 Function implementation

In an OOP simulation model, class functions are implemented to calculate objective and constraint metrics. An important class function for enforcing the design constraints of the CMM is the pillar motion resolution calculation. The pillar motion resolution is each pillar's required linear actuator resolution to achieve the end effector motion resolution requirement. Given the target build envelope radius, the end effector pose resulting in the worst-case pillar

resolution value can be determined. This value is returned by the class function to be compared to the linear actuator design constraint.

Furthermore, a multilateration function was implemented to localise the touch probe in the work envelope. Multilateration is a geometric technique to determine an object's position by measuring its distance from multiple known reference points. The multilateration algorithm, derived by Norrdine [13], was translated into Python for use in this study. The 3 linear sensor measurements and 3 sensor reference points are used in this implementation. Utilizing 3 reference points and their distance measurements typically results in 2 possible solutions. However, one solution is consistently eliminated because the entire work envelope lies below the reference points.

Based on this multilateration algorithm, a function was implemented to quantify the position error resulting from linear measurement errors in the sensors. The linear error of each laser sensor is represented as a bidirectional error value quantified as Parts Per Million (PPM) relative to the measured distance. Based on different combinations of over- or under-estimations of the 3 sensors, a confidence volume can be created for a given position of the touch probe. The multilateration error can then be calculated as the maximum distance from the probe position to an infliction point on the surface of the confidence volume.

Finally, a function was implemented to calculate the maximum multilateration error in the work envelope. This is required optimise the machine's measurement accuracy. The work envelope was divided into a grid of 3D points, using small intervals of radius, height, and z-angle from the machine's origin. The multilateration error is then calculated at each point of the grid, and the maximum value is returned as the machine's maximum multilateration error.

2.2 Multi-objective problem definition

The multi-objective problem is defined by identifying parameters and metrics to be used as objectives, constraints, and decision variables. This is done by translating the traditional design requirements and constraints into a multi-objective optimisation context.

2.2.1 Objective identification

The objectives represent the goals to be simultaneously minimized or maximized during the optimisation process. Table 1 describes the parameters and metrics optimised in the context of optimising the CMM design.

Table 1. Description of optimisation objectives.

Parameter or metric	**Description**
Maximum multilateration position error	The worst-case multilateration error in the work envelope resulting from distance sensor measurement errors. This is minimized to ensure maximum machine measurement accuracy.
Pillar motion resolution	Calculated worst-case pillar motion resolution required to satisfy the 3D motion resolution design requirement of the touch probe in the work envelope. This is maximized to lower the cost and difficultly of machine construction, since higher resolution motion systems are increasingly expensive and complex.
Sensor arm horizontal length	The horizontal length of each arm that mounts the distance sensors is minimized to decrease machine size.
Sensor arm vertical length	The vertical length of each arm that mounts the distance sensors is minimized to decrease the complexity of the machine, because the sensor arms are mounted to the roof plate and longer members are increasingly complex to mount rigidly.

Distance sensor stroke length	The distance sensors stoke distance required to reach the extreme positions of the work envelope is minimized to decrease the complexity of the machine, because a longer extension mechanism is heavier and more prone to bending.
Base radius	The distance from the base plate origin to each pillar mount is minimized to decrease the size of the machine.
Pillar height	The length of the pillars is minimized to decrease the size of the machine, which also increases rigidity.

2.2.2 Constraint identification

Constraints are conditions or limitations that restrict the feasible solution space during the optimisation process. In the context of the CMM design, each of the objectives listed in Table 2 are also constrained based on relevant design requirements. Table 2 only describes the additional parameters and metrics that are constrained and are not included as objectives. Note that constraints on decision variable values are not applied as optimisation constraints but as an input range used by the genetic algorithm sampling algorithm.

Table 2. Description of optimisation constraints excluding those listed as objectives.

Parameter or metric	**Description**
Impossible build check boolean	The CMM model includes a function to verify that the combination of decision variables does not result in an impossible machine. This condition is included as a constraint.
Arm vertical angle	The vertical angle between each arm and corresponding pillar is constrained to an upper and lower angle to prevent a singularity posture in which the machine is locked.
Arm joint out-of-plane angle	The arms are joined to the linear actuators with a ball joint. The out-of-plane angle is constrained based on the best-case ball joint specifications in available catalogues.
Pillar stroke length	The stroke length of the linear actuators on the pillar required to serve the extreme points of the work envelope. This is constrained based on the best-case units in available catalogues.
Maximum distance sensor length	The maximum length of the distance sensor unit to reach the extreme points of the work envelope is constrained based on a design requirement.

2.2.3 Decision variable selection

The identification of decision variables is essential for formulating the optimisation problem and determining the parameters to be adjusted during the optimisation process. Firstly**,** design parameters that directly influence the objectives and constraints described in previous sections are identified. This includes geometric dimensions, actuator specifications, and control parameters. The design parameters that do not impact the optimisation problem are given a fixed value before the simulation. For example, the distance between each parallel rod connecting the end effector to a pillar influences the stability of the end effector but does not affect any identified objectives or constraints. Therefore, this value can be fixed before the optimisation.

Finally, the remaining design parameters are identified as decision variables. These are the variables that will be manipulated by the optimisation algorithm to explore the design space. For the design of the CMM, the final decision variables were identified as, sensor arm horizontal length, sensor arm vertical length, base radius, and arm length.

2.3 Optimisation algorithm execution

Genetic algorithms are well-suited for a generalised optimisation strategy due to their ability to handle non-linear, complex, or discontinuous models without requiring explicit mathematical expressions. For this study, the NSGA-II [1] algorithm was selected, which is a genetic algorithm widely used in research [2]. The Python library, Pymoo [12], is used to implement the NSGA-II algorithm due to its flexibility and ease-of-use. Fig. 5 shows a code snippet of how the NSGA-II optimisation algorithm is initialised and executed. This is the entry point script is run to start an optimisation run.

```python
from pymoo.algorithms.moo.nsga2 import NSGA2
from delta_robot_accuracy_problem import DeltaRobotProblem
from pymoo.termination.max_gen import MaximumGenerationTermination

problem = DeltaRobotProblem()
algorithm = NSGA2(pop_size=1000)
algorithm.setup(
    problem=problem,
    termination=MaximumGenerationTermination(1000),
    seed=1,
    verbose=True
)

while algorithm.has_next():
    algorithm.next()

results = algorithm.result()
```

Fig. 5. Code snippet of NGSA-II algorithm execution.

The only hyperparameter modified from its default value is the population size, which is selected as 1000. This was selected after experimenting with a range of values to determine a population size small enough for quick optimisation runs, and large enough to consistently converge to global Pareto optimal solutions. The algorithm is set to terminate after an excessively high generation count of 1000 to ensure the algorithm does not terminate prematurely. The algorithm can be manually terminated early if convergence is reached. Convergence is determined by monitoring the epsilon metric after each cycle of the genetic algorithm until it reaches steady-state. Pymoo calculates this metric based on the hypervolume of the Pareto front.

```python
import numpy as np
from pymoo.core.problem import ElementwiseProblem
from delta_robot import DeltaRobot

class DeltaRobotProblem(ElementwiseProblem):
    def __init__(self):
        super().__init__(
            n_var = 4, # Number of decision variables
            n_obj = 7, # Number of objectives
            n_ieq_constr = 11, # Number of inequality constraints

            # Upper and lower bounds of decision space:
            xl = np.array([  0.0,   0.0, 200.0, 200.0]),
            xu = np.array([600.0, 600.0, 600.0, 1000.0])
        )

    def _evaluate(self, x, out, *args, **kwargs):
        robot: DeltaRobot = DeltaRobot(
            sensor_arm_horizontal_length = x[0],
            sensor_arm_vertical_length   = x[1],
            base_radius                  = x[2],
            arm_length                   = x[3],
        )
        out["F"] = self.get_objectives(x, robot)
        out["G"] = self.get_constraints(out["F"], robot)
```

Fig. 6. Code snippet of the class definition of DeltaRobotProblem.

The optimisation problem is defined in the custom Python class called DeltaRobotProblem, which inherits the Pymoo ElementwiseProblem class. Fig. 6 shows a code snippet of how the DeltaRobotProblem class is defined. The OOP structure of our model forces us to evaluate one solution at a time. The ElementwiseProblem enables this, by allowing each solution to be evaluated independently, in contrast to other problem classes which evaluate multiple solutions at once using matrix operations.

DeltaRobot is the class that defines the simulation model and kinematics. During optimisation, the "_evaluate" function is iteratively called by Pymoo to evaluate the objectives and constraints of each solution in the population. The objectives vector and constraints vector of each solution is returned in out["F"] and out["G"] respectively.

The function, "get_objectives", calls various functions from the simulation model in the DeltaRobot class to return a vector of values representing the optimisation objectives. Likewise, the "get_constraints" function returns a vector of values representing the evaluated constraints. The Pymoo module then uses the NSGA-II algorithm to minimize the objective vector values, while satisfying the inequality constraint that each constraint vector value should be smaller than zero.

2.4 Results visualisation

Visualization of optimisation results helps the designer understand the relationships between design variables and objectives. Effectively visualizing high-dimensional data is challenging because a limited number of variables can be compared in a single plot. Four variables can be compared in a single 3D scatterplot by assigning each variable to the x, y, and z axes, and utilizing a colour scale to represent the fourth variable. In this plot, each point in the scatter plot represents a non-dominated solution on the Pareto front.

An interactive visualisation dashboard was developed to streamline the analysis of the results. Fig. 7 shows a screenshot of the dashboard software. The dashboard enables the designer to upload results as a comma-separates values (CSV) file. Variables for each axis and colour scale can then be selected interactively and the results can be filtered based on chained inequality filters.

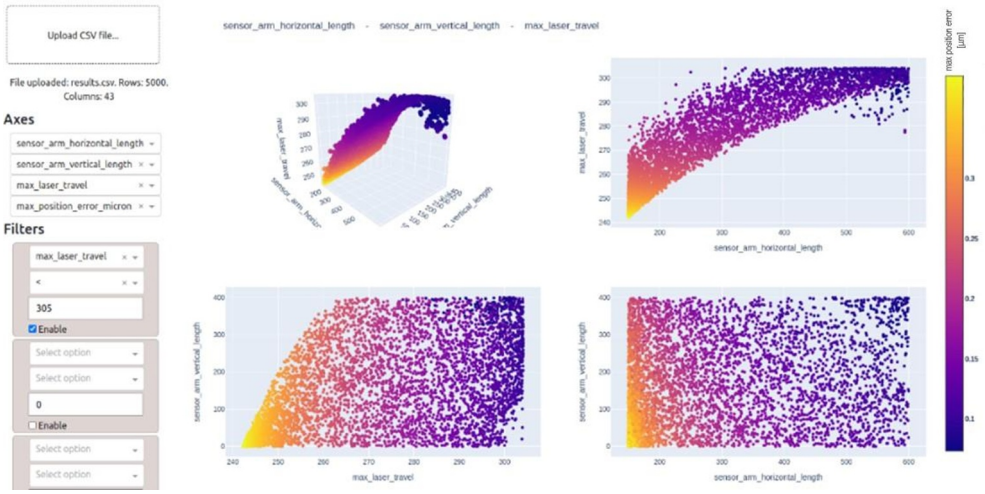

Fig. 7. A screenshot of the visualisation dashboard with data displayed in plots.

2.5 Design decisions

The results visualisation presents the Pareto optimal solutions to the designer. Due to the nature of the Pareto front, no single solution can be deemed optimal. Therefore, design intuition needs to be applied to reduce the solution space to a single design.

In this phase of the process, a design decision is applied by fixing the value of a decision variable or by adding a constraint. Each decision is applied sequentially, refining the optimisation problem to enable iterative reruns with a diminished solution space. This is repeated until all decision variables are fixed.

Typically, when an additional constraint is applied, it constrains a dependant variable derived from decision variables. For example, if the visualisation exposes an inflection point where an increase in sensor stroke distance yields marginal gains in other objectives, a constraint is applied at this inflection point. Likewise, a decision variable is fixed at a point which negligible gains in other objectives are observed. Finally, a singular solution can be selected from the reduced solution space, where opting for an alternative solution would offer no practical advantage.

2.6 Sensitivity analysis

The evaluation of how sensitive an objective is to small changes in a design parameter is an important aspect of sensitivity analysis in multi-objective optimization problems. This step is required for validating the stability and robustness of the optimization outcomes for practical applications. If an objective is very sensitive to small changes in a design parameter, parameter changes like manufacturing tolerances or temperature expansion may result in the manufactured machine deviating significantly from the optimal performance predicted.

To perform this sensitivity analysis on an objective and design parameter pair, the parameter value is increased by a small increment, the objective is recalculated, and the deviation from the original value is recorded. The parameter value is then decreased by the same amount and the corresponding deviation in the objective is recorded. Only the maximum objective deviation value needs to be considered. This process is then repeated for every solution on the Pareto front, and every related design parameter and objective, to determine which solutions are too sensitive to be considered in a practical application. Solutions with larger deviation values are deemed more sensitive, and likewise, solutions with smaller deviations are deemed more robust.

3 Results

This section explores the results of applying the design optimisation methodology from the previous section to the design of the CMM. Fig. 8 and Fig. 9 plot the Pareto front solutions of the optimisation problem as described in the previous section. The pillar motion resolution represented in these plots is the linear actuator resolution required to achieve an end effector resolution of 0.5 μm in the work envelope. A large pillar motion resolution is preferred because it simplifies the manufacturing process. Furthermore, a small base radius is desirable to reduce the overall size of the machine. Based on design intuition, a base radius value of 400 mm is selected and fixed, because a larger base radius becomes impractical and does not offer significant gain in objectives. The corresponding pillar resolution of 0.10 to 0.17 μm is deemed to be achievable.

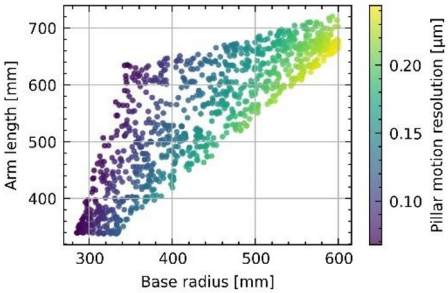

Fig. 8. Pareto front solutions of first optimisation comparing base radius to arm length.

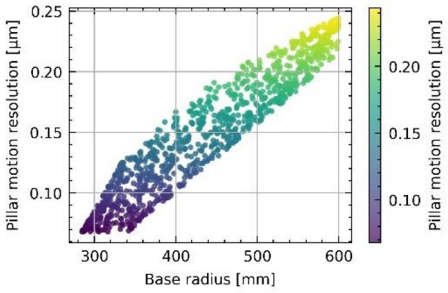

Fig. 9. Pareto front solutions of first optimisation comparing base radius to pillar motion resolution requirement.

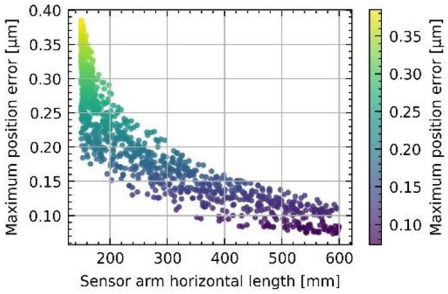

Fig. 10. Pareto front solutions of second optimisation comparing sensor arm horizontal length to maximum position error.

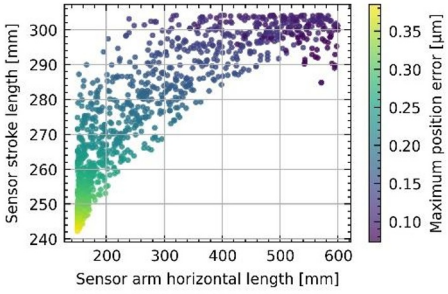

Fig. 11. Pareto front solutions of second optimisation comparing sensor arm horizontal length to sensor arm vertical length.

Fig. 10 and Fig. 11 show the Pareto front solutions after the base radius value was fixed and the optimisation was run again. The maximum position error plotted is defined as the maximum measurement error at any point in the work envelope due to linear measurement errors in the CMM. The horizontal length of the sensor arm is fixed at 400 mm, as extending this length further does not yield a significant improvement in measurement accuracy.

Fig. 11 shows the minimum stroke length of the distance sensor increases for higher horizontal lengths of the sensor arm. However, a stroke length of more than 284 mm is satisfactory, which corresponds to the horizontal sensor length fixed at 400 mm.

Fig. 12 and Fig. 13 plot the Pareto optimal solutions with the arm length and sensor vertical length being the only free decision variables in the optimisation. Both Fig. 12 and Fig. 13 show a small outlier cluster of solutions with a sensor height below 400 mm. The outlier cluster has a low sensor height, which results in a shallow vertical angle of the sensor line of sight to the top of the work envelope. This corresponds to high accuracy in the horizontal plane but a low accuracy in the vertical axis, which is undesirable. Therefore, a constraint is employed to exclude the outlier cluster and constrain the sensor height above 500 mm.

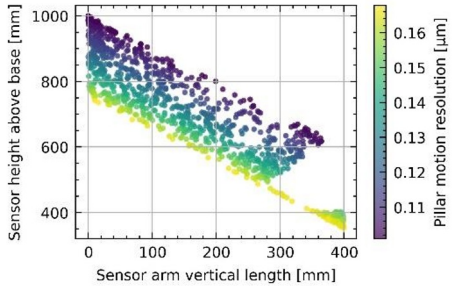

Fig. 12. Pareto front solutions of third optimisation comparing sensor arm vertical length to sensor height above base.

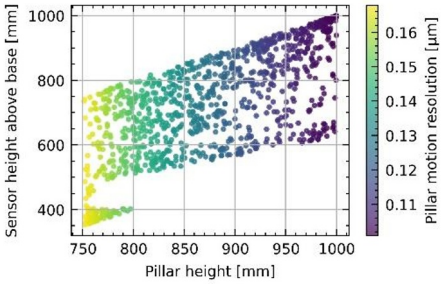

Fig. 13. Pareto front solutions of third optimisation comparing pillar height to sensor height above base.

Fig. 14, Fig. 15, Fig. 16, and Fig. 17 plot the Pareto optimal solutions after the additional constraint was added to exclude the outlier cluster and reduce the input range of the sensor height. In Fig. 14, the colour scale shows that a shorter arm length corresponds to a better pillar motion resolution. Furthermore, the plot shows that solutions with a maximum position error below 0.12 µm exist for the entire arm length range. Hence, the arm length does not significantly limit the position error objective. Fig. 15 shows that the sensor stroke length values in the given solution set fall between 280 and 305 mm. Even though minimizing the sensor stroke length is an optimisation objective, the small difference between the minimum and maximum values will have a negligible effect on the practical design.

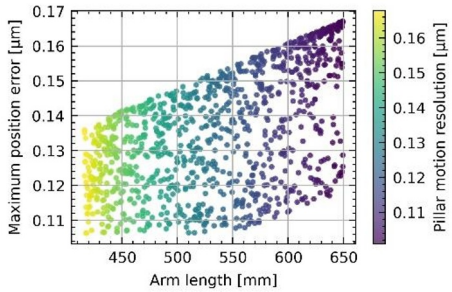

Fig. 14. Pareto front solutions of fourth optimisation comparing arm length to maximum position error.

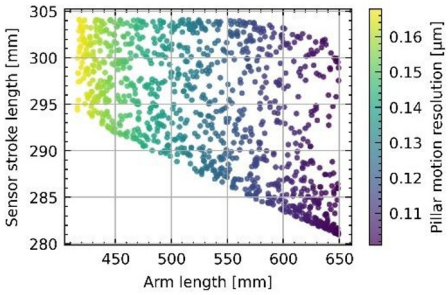

Fig. 15. Pareto front solutions of fourth optimisation comparing arm length to sensor stroke length.

Fig. 16 shows the correlation between the sensor arm vertical length and the maximum position error in the work envelope. Note that the position error values fall between 0.10 and 0.17 µm which is well within the design requirement of 1 µm and is negligibly small range. Furthermore, Fig. 17 shows a positive correlation between the arm length and pillar height of the CMM. All these solutions are in the Pareto optimal solution set and are suitable choices for the final design.

Finally, the solution set has been sufficiently narrowed for the design engineer to make the final parameter decisions during the final drafting process, utilizing discrete parameters from off-the-shelf components. Qualitative objectives, such as ease of manufacturability, often have a greater significant impact on the quality of the machine than marginal improvements in quantitative metrics at this stage of the design.

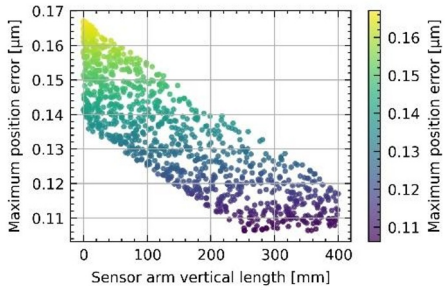

Fig. 16. Pareto front solutions of fourth optimisation comparing sensor arm vertical length to maximum position error.

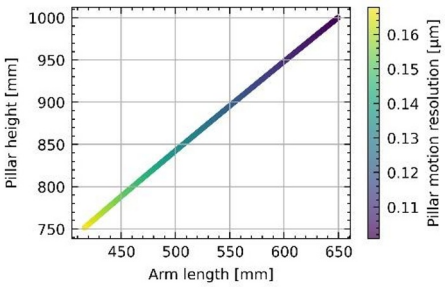

Fig. 17. Pareto front solutions of fourth optimisation comparing arm length to pillar height.

A sensitivity analysis of the final solution space ensures that all considered solutions are robust enough for a practical application of the CMM. This is to avoid selecting a sensitive solution that yields suboptimal results in a practical setting due to small changes in parameter values that significantly deteriorate the design objectives. Fig. 18, Fig. 19, Fig. 20, and Fig. 21 shows scatter plots of the different solutions on the Pareto front from the final optimisation run. Fig. 18 shows the maximum deviation in the maximum position error objective due to a 1 mm increase or decrease in the sensor arm horizontal length. Likewise, Fig. 19 shows the maximum deviation in the objective due to a 1 mm change in the sensor arm vertical length. A larger deviation value of an objective corresponds to a more sensitive solution.

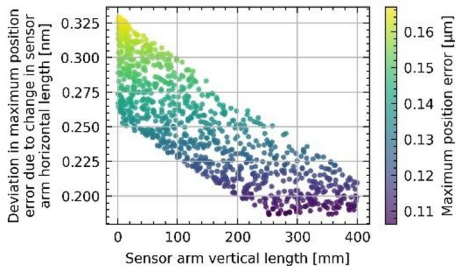

Fig. 18. Pareto front solutions of fourth optimisation showing deviation in maximum position error due to 1 mm change in sensor arm horizontal length.

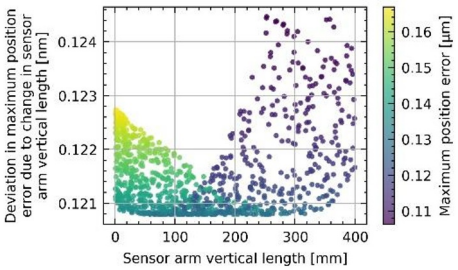

Fig. 19. Pareto front solutions of fourth optimisation showing deviation in maximum position error due to 1 mm change in sensor arm vertical length.

Recall that in this solution space, the sensor arm horizontal length is fixed, but the sensor arm vertical length is a decision variable. In both plots the *x*-axis represents the sensor arm vertical length. Fig. 18 shows that the sensitivity to changes in the sensor arm horizontal length decreases with an increasing sensor arm vertical length. However, Fig. 19 shows that the sensitivity to changes in the sensor arm vertical length is highest for longer sensor arm vertical lengths. The highest deviation value in the maximum position error objective in both plots is 0.33 nm, which is negligibly small.

Fig. 20 shows the maximum deviation in the pillar motion resolution objective due to a 1 mm increase or decrease in the base radius. Likewise, Fig. 21 shows the maximum deviation in the objective due to a 1 mm change in the arm length. Both plots have a similar trend, with sensitivity decreasing with an increasing design parameter value. The highest deviation in the pillar motion resolution objective from both plots is 1.15 nm, which is negligible for a

practical machine. Hence, the final solution space is robust to changes in design parameters, and the designer can continue to select a solution from this set.

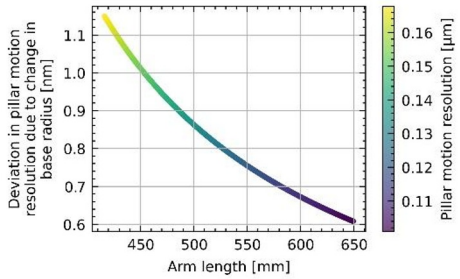

Fig. 20. Pareto front solutions of fourth optimisation showing deviation in pillar motion resolution due to 1 mm change in base radius.

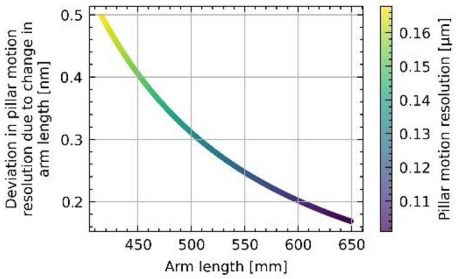

Fig. 21. Pareto front solutions of fourth optimisation showing deviation in pillar motion resolution due to 1 mm change in arm length.

4 Conclusion

In this study, a systematic and generalisable framework for multi-objective optimisation in machine design was presented and demonstrated through the optimisation of a delta CMM. Utilizing the NSGA-II algorithm, the Pareto optimal solutions were identified that balance critical design objectives such as measurement accuracy, motion resolution, and machine size.

The results demonstrate the efficacy of the proposed methodology in addressing the complexities of MOO problems in machine design. Through an iterative process, involving repeated simulation-guided parameter refinements and decision-making steps, a small solution set was reached that provides a designer with a narrow range of design parameters that can be further refined based on qualitative design objectives. A sensitivity analysis also proved that the objectives of the optimal solutions were robust enough for a practical application of the CMM.

Despite the robustness and generalisability of the presented framework, it is important to acknowledge the important role of design intuition. Qualitative aspects of design problems, such as user preferences and manufacturability, often cannot be fully quantified in an optimisation model. Therefore, the integration of qualitative objectives and intuition remains an essential part of the final decision-making process to ensure that the optimised solutions are practically viable and aligned with broader project goals.

Overall, a systematic and generalisable framework for multi-objective design optimisation was successfully demonstrated. The framework provides a reusable methodology for machine design optimisation while leaving enough flexibility for informed decision-making based on qualitative objectives and design intuition.

References

1. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, IEEE Trans. Evol. Comput., **6**, 2, pp. 182–197, (2002)
2. J. L. J. Pereira, G. A. Oliver, M. B. Francisco, S. S. Cunha, and G. F. Gomes, Arch. Comput. Methods Eng., **29**, 4, pp. 2285–2308, (2022)
3. Q. Ma, A. EL-Refaie, and A. Fatemi, *Multi-objective Design Optimization of a Blended Permanent Magnet Assisted Synchronous Reluctance Machine*, in *2021 IEEE International Electric Machines & Drives Conference, IEMDC*, pp. 1–7, (2021)
4. Y. Yan, D. Guangming, C. Liang, Z. Chong, and P. Lei, *Robust design optimization based on multi-objective particle swarm optimization*, in Proceedings of the *IEEE Congress on Evolutionary Computation (CEC)*, pp. 4918–4925 (2016)
5. A. Ayaz and A. Rasheed, *Multi-Objective Design Optimization of Multicopter using Genetic Algorithm*, in Proceedings of 18th International Bhurban Conference on Applied Sciences and Technologies, IBCAST, Jan. 2021, pp. 177–182, (2021)
6. P. Lutat, L. Molitor, M. Kurth, A. Würde and A. Ulbig, *Physics-Aware Planning of Broadband Powerline Communication in Low Voltage Distribution Grids*, in Proceedings of the International Conference on Smart Energy Systems and Technologies, SEST, Mugla, Turkiye, pp. 1-6, (2023)
7. S. Hammad Mian and A. Al-Ahmari, New developments in coordinate measuring machines for manufacturing industries, Int. J. Metrol. Qual. Eng., **5**, 1, pp. 1–10, Sep. (2014)
8. G. Venter, *Review of Optimization Techniques*, in Encyclopedia of Aerospace Engineering, Wiley, (2010)
9. C. Wampler, IEEE Trans. Syst. Man. Cybern., **16**, 1, pp. 93–101, (1986)
10. G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, (2009)
11. C. R. Harris *et al.*, Nature, **585**, 7825, pp. 357–362, (2020)
12. J. Blank and K. Deb, *IEEE Access*, **8**, pp. 89497–89509, (2020)
13. A. Norrdine, *An algebraic solution to the multilateration problem*, in Proceedings of the 15th international conference on indoor positioning and indoor navigation, Sydney, Australia, vol. 1315, (2012)