PRODUCT PLATFORM AUTOMATION FOR OPTIMAL CONFIGURATION OF INDUSTRIAL ROBOT FAMILIES

Mehdi Tarkian[1], Johan Ölvander[1], Xiaolong Feng[2] and Marcus Pettersson[2]
(1) Linköping University, Sweden
(2) ABB Corporate Research, Sweden

ABSTRACT

Product platform design is a well recognized methodology to effectively increase range and variety of products and simultaneously decrease internal variety of components by utilizing modularization. The tradeoff between product performance and product family commonality has to be carefully balanced in order to for the company to meet market requirements and simultaneously obtain economy of scale. This paper presents a framework based on high fidelity analyses tools that concurrently optimize an industrial robot family as well as the common platform. The product family design problem is formally stated as a multi-objective optimization problem, which is solved using a multi-objective Genetic Algorithm.

Keywords: Automated design, multidisciplinary design optimization, parametric CAD

1. INTRODUCTION

Product family design based on modularization has for a long time been a well recognized method to address the demands of mass customization [1]. Based on the concept of product platforms, it is possible to deliver products within a short time frame and have a broad product range to meet specific customer requirements while maintaining low development and manufacturing costs [1]. A potential drawback of product families is that the performance of individual members are reduced due to the constraints added by the common platform, i.e. parts and components need to be shared by different family members.

This paper focuses on quantitative approaches where the product family design problem is formally stated as an optimization problem where high fidelity analyses tools are used to find a tradeoff between degree of commonality and product performance. The optimization problem involves balancing performance of the members of the product family against cost savings during design and possible re-use of modules in the family.

The outline of the paper is as follows: Following the introduction, section 2 explains the scope of the paper by describing the concept of product family and platform design and the research conducted in the global research arena. A brief outline of how the identified obstacles ought to be tackled is given. In section 3, an overview of the field of Multidisciplinary Design Optimization (MDO) and Knowledge Based Engineering (KBE) is presented. These techniques are two important enablers to pursue practical product family design and optimization. The design procedures and hurdles of modular industrial robot design are presented in section 4. The automated framework along with the optimization procedures adopted is described in section 5. Finally, in section 6 the paper is concluded.

2. PLATFORM DESIGN

The definitions of product platform are plenty and since it is a fundamental term in this paper the following definition has been chosen; *"the use of a standard module set between different products is known as a platform"* [2]. Thereby, a platform is the set of standard components, manufacturing processes, and/or assembly steps that are common in a set of products. The overall aim with product family design is to reduce cost due to the commonality between the variants. However, there is always

a trade-off between commonality and performance of individual family members [3] & [4].

There are many benefits or reasons for modularity, e.g. the twelve modular drivers described in [5]. However, most of them have economic implications, either in the design and development stage, in purchasing, during manufacturing or in the aftermarket. In the literature there are many indices defined to measure the degree of modularity within a product family, see [6]. A commonality index is typically based on different parameters such as the number of common components, the component manufacturing volume, the component costs, the manufacturing processes, and so on. The leading principle of the indices is to provide an estimate of the cost savings within a family.

2.1. Modularization

Jose and Tollenaere [7] describe modularization as *"an approach to organize complex designs and process operations more efficiently by decomposing complex systems into simpler portions"*. Modularization as a way to save cost is by no means a new phenomenon. For instance, the truck manufacturer Scania has been working successfully with modularization and the concept of product families since the forties [1]. Much research in product family design has been qualitative to its nature. Hence according to Jose and Tollenaere [7], *"today the methods for platform product development are not practical and future results can be obtained with an integral methodology using a practical design representation linked to an optimization methodology"*. Thus, this paper makes an effort to presents a quantitative approach where product family design is formulated as a formal multi-objective optimization problem.

Before initiating the modular design process, the product has to be evaluated for whether being appropriate for modular design. The modularity level of the product is then determined and strategies for modular design are carried out. To assure practical use of the outlined framework in industry, tight interaction with commercial CAE and simulation tools are of greatest importance. In order to accomplish system optimization incorporating various simulation tools, the concept of Multidisciplinary Design Optimization has been adopted, which will be discussed in the following section.

3. MULTIDISCIPLINARY DESIGN OPTIMIZATION

Li and Huang [8] recognize that by using different platform strategies such as commonality, modularity and scalability, product platforms can be developed and customized with different flexibility for realizing *mass customized products* [8]. This enabler is termed *adaptive platform*. Li and Huang also coined the terms *scalable modules* and *instance module*, established to achieve adaptive platforms. Furthermore, in the design of complex and tightly integrated engineering products it is essential to be able to handle cross-couplings and synergies between different subsystems [9]. A typical example of such products is mechatronic machines like industrial robots. To effectively design and develop such products, efficient tools and methods for integrated and automated design are needed throughout the development process. Multidisciplinary Design Optimization (MDO) is one promising technique that has the potential to drastically improve such a concurrent design. Giesing et al. [10] have defined MDO as *"a methodology for the design of complex engineering systems and subsystems that coherently exploits the synergism of mutually interacting phenomena"*.

3.1. Knowledge Based Engineering for Design of Engineering Products

Knowledge-Based Engineering defines a wide range of methods and processes and could be described in several ways depending on the application focus. In the literature there are various definitions that strive to highlight the multiple sides of KBE. Chapman et al. [11] define KBE as "*an engineering method that represents a merging of object oriented programming (OOP), artificial intelligence (AI) techniques and computer-aided design technologies, giving benefit to customized or variant design automation solutions*". It therefore presents great potential for improving the product development process as well as reducing the time-to-market, thanks to an enhanced effectiveness of computer aided engineering systems.

It is the authors' opinion that KBE is a means to achieve design reuse and automation, and thereby create prospects for a holistic perspective throughout the design process.

A holistic product perspective by means of design reuse and automation is needed in order to effectively manage product complexity and to introduce MDO. In this field, KBE is believed to be a powerful tool [12]. In the coming sections methods for design automation are proposed.

3.2. Dynamic Top-Down Modeling

By introducing KBE techniques a new mean of CAD modeling is introduced, referred to here as dynamic top-down modeling. When applying a dynamic top-down development process, the actual CAD models can be generated from pre-described High Level CAD templates (HLCt). The critical information on how the HLCt should be instantiated is stored in the inference engine [14]. The geometry model is divided into sub-models that are linked to each other in a hierarchic relational structure [15]. Various components can be attached dynamically to the model and their shape altered by the inherited design variables, supporting the concept of *scalable modules* by Li and Huang [8]. This process continuous until the geometry is completely defined.

4. DESIGN APPLICATION: INDUSTRIAL ROBOTS

The mechanical structure of a modular industrial robot consists of a base followed by a series of modular structure links. Each module consists of drive-train components (servo actuator, combining precision Harmonic Drive gearing with highly dynamic servo motors). Major components of the robot controller are power units, rectifier, transformer, axis computers and a high level computer for motion planning and control.

Designing industrial robots is a complex process involving tremendous modeling and simulation effort. For all the various domains of robot design, the geometry plays a significant role as input provider.

To more effectively understand and manage the complexity of this technology and find the optimal solution for a family of robots faster, a joint novel design framework is being developed at ABB and Linköping University, see Johansson et al. [16], Petterson et al [17], and Tarkian et al. [13].

4.1. Modular Geometry approach for Modular Industrial Robots

One outcome of modularity within a product family is increase of variety and decrease of components. The same principle is adopted here for the modeling of the product family. Since the geometries are saved as HLCts and instantiated with unique internal design variables, the number of model variants is effectively increased by sharing few geometric templates between the model variants.

By importing the HLCt geometries, the robot is defined in three steps, see Figure 1. Firstly the number of axes is determined in a user interface, defining the skeleton model of the robot, stored in the Datum HLCt and placed according to the logic of the inference engine. The type of Component HLCt for each axis is then decided and an appropriate structure, from Structure HLCt, is chosen in the final step. The model of the robot is thereby transformed from an empty initial model into a complete model in three steps, as shown in Figure 1.

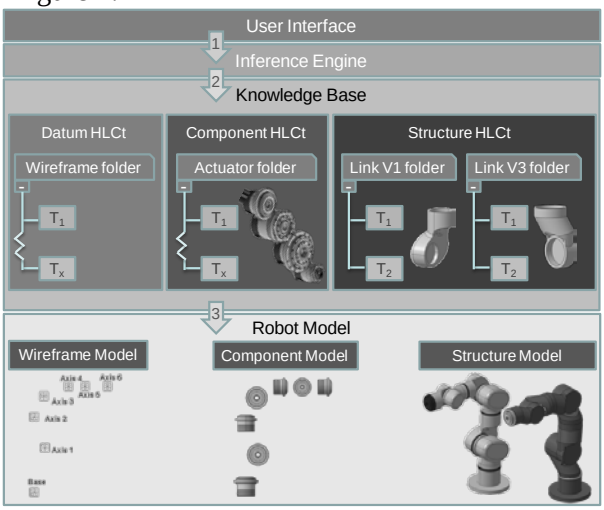

Figure 1. Relations between the robot models and the HLCt libraries.

4.2. Dynamic Model

To simulate the dynamic properties of a robot, a dynamic model has to be utilized. The dynamic model in the outlined framework is made using an in-house simulation tool developed at ABB. The motion of the rigid manipulator can be described by

$$Q = M(q)\cdot\ddot{q} + V(q,\dot{q}) + G(q) + B(\dot{q}) \qquad (1)$$

where M is the inertia matrix, V is the vector of Coriolis and centrifugal forces, G is a vector of gravity forces and B is a vector of viscous friction forces. q is a vector of generalized coordinates e.g. angular position of each joint in the manipulator. For more information about dynamic models and trajectory planning for robots see [18] & [19].

In the Newton-Euler formulation [18], link velocities and acceleration are iteratively computed, forward recursively.

$$\begin{aligned} a_{e,i} &= \left(R_{i-1}^{i}\right)^{T} a_{e,i-1} + \dot{\omega}_i \times r_{i,i+1} + \omega_i \times \left(\omega_i \times r_{i,i+1}\right) \\ a_{c,i} &= \left(R_{i-1}^{i}\right)^{T} a_{e,i-1} + \dot{\omega}_i \times r_{i,ci} + \omega_i \times \left(\omega_i \times r_{i,ci}\right) - \left(R_{0}^{i}\right)^{T} g_0 \end{aligned} \qquad (2)$$

When the kinematic properties are computed, the force and torque interactions between the links are computed backward recursively from the last to the first link.

$$\begin{aligned} f_i &= R_i^{i+1} f_{i+1} + m_i a_{c,i} \\ \tau_i &= R_i^{i+1}\tau_{i+1} - f_i \times r_{i,ci} + \left(R_i^{i+1} f_{i+1}\right) \times r_{i+1,ci} + I_i\alpha_i + \omega_i \times \left(I_i\omega_i\right) \end{aligned} \qquad (3)$$

Where ω is the angular velocity and $\dot{\omega}$ angular acceleration, a_e and a_c describe the acceleration at the end and at the center of each link respectively, f and τ describe the force and torque between each link respectively. R is the rotational matrix, I the mass inertia, g_0 the gravity acceleration and r the positional vector.

4.3. Automated and Holistic Design approach

The geometric and dynamic models are seamlessly integrated through a user interface, where various engineering aspects of the robot are analyzed concurrently. Furthermore the geometrical and dynamical aspects of the robot components are stored in a component library, see Figure 2.

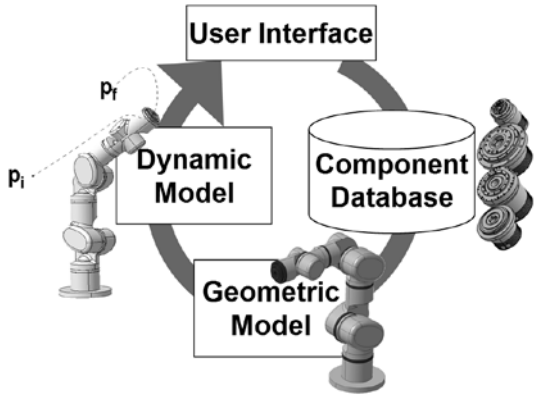

Figure 2. Weight and dynamic properties are concurrently computed following parametric input in the user interface.

5. OPTIMIZATION

In this section the problem formulation is presented following the selection of optimization algorithm for the specified problem, as well as the actual optimization framework.

5.1. Problem Formulation

The problem formulation consists of concurrently optimizing the performance and commonality level of a product family consisting of four robots. The optimization variables are choice of servo actuators for axes 1, 2 and 3 as well as a coefficient defining the relation between length of link 2 and

link 4 (see figure 3), amounting to overall 16 optimization variables for the entire product family. The optimization problem is limited to three axes in order to restrict the design space for the optimization algorithm. Ideally all 7 joint could be optimized using the same approach depicted. The four robots' reach and payload requirements are visualized in Table 1.

	Robot 1	**Robot 2**	**Robot 3**	**Robot 4**
Reach [mm]	760	860	960	1060
Payload [kg]	3	5	10	14

Table 1. Payload and reach requirements of the robot family

The problem is multi objective with the performance and commonality being the objective functions. The performance objective, f_1, is the sum of cycle time (CT) and the robot weight (Weight) for all four robots. The performance objective is to be minimized, hence low weight and low cycle time is preferred. The commonality objective is to maximize number of common components in the robot family, for both the links and actuators. f_2 is the percent commonality, ranging from 0 to 100.

$$f_1 = \sum(\lambda_1 CT_i + \lambda_2 Weight_i)$$
$$f_2 = 100 \cdot (k_1 \frac{\sum Link_{shared}}{\sum Link} + k_2 \frac{\sum Actuator_{shared}}{\sum Acutator}) \qquad (4)$$
$$i = 1,2,3,4$$

λ_j & k_u are weighting factors where $\sum k_u = 1$. The weighting factors k_u have been chosen to prioritize link share prior to actuator share. The weight and CT are normalized and λ_j chosen to balance the weighting.

5.2. Multi-Objective Genetic Algorithm

The presented problem consists of discrete variables, and the objectives and constraints are represented by non-linear functions where no analytical derivatives are available. Therefore a Genetic Algorithm has been chosen since generally speaking non-gradient methods are applicable to a broader range of problems as they do not rely on assumptions on the properties of the objective function such as differentiability and continuity, etc. The basic idea of Genetic Algorithms is the mechanics of natural selection [20]. Each optimization variable is coded into a gene as for example a real number or a string of bits. The corresponding genes for all parameters form a chromosome, which describes each individual. Each individual represents a possible solution, and a set of individuals form a population. In a population, the fittest individuals have the highest probability of being selected for mating. Mating is performed by combining genes from different parents to produce a child, called a crossover. Then there is also the possibility that a mutation might occur. Finally the children are inserted into the population to form a new generation.

Moreover since the tradeoff between performance and commonality is difficult to quantify beforehand, preferably the algorithm utilized should generate a Pareto frontier of the design solutions. Optimization methods that can handle this type of problems in general are Genetic Algorithms and specifically Multi-Objective Genetic Algorithms [21]. There are also many examples in the literature where GA:s and MOGA:s are applied to platform design problems [22]& [23]. In this paper NSGA-II are used as the optimizer [24].

5.3. Optimization framework

In previous work [13], a robot design framework has been utilized to design a single optimal modular robot for a specific task and a set of requirements. A product family optimization requires further evaluations to converge since the family members are simulated in sequence. Also, to reach convergence, the number of evaluations increases due to increased number of design variables. Consequently, to shorten the optimization time, the earlier framework [13] had to be completely reworked, which will be further depicted in following sections.

5.3.1. Geometry Database

Although commercial CAD tools are well suited to generate high fidelity geometry for various

analyses tools, they often require extensive update time. Therefore, a geometry database has been created to eliminate the lengthy simulation times required. The database is created by evaluating and storing an array of various geometric configurations. Meaning that the shape and number of the robot structure are varied leading to a new robot configurations of which the geometric properties are stored as illustrated in Figure 3. The geometric properties include mass, center of gravity and inertia.

In Figure 3 all links subjected to parametric modification are colored white. For link 1 and link 3 the shape is altered by modifying the type of actuator. These are modified by altering discrete values ranging from 1 to 15 which will automatically insert the actual detailed actuator geometry which is stored as an HLCt. The logic stated in the inference engine will then update the internal design variables of the links housing the actuator. For link 2 and link 4 the shape of the structure is modified by varying the lengths between 200-450 [mm] and 200-400 [mm] respectively.

The geometry database for each link is computed independently. The separate mass properties of the links are then assembled together during the optimization framework to represent the complete robot.

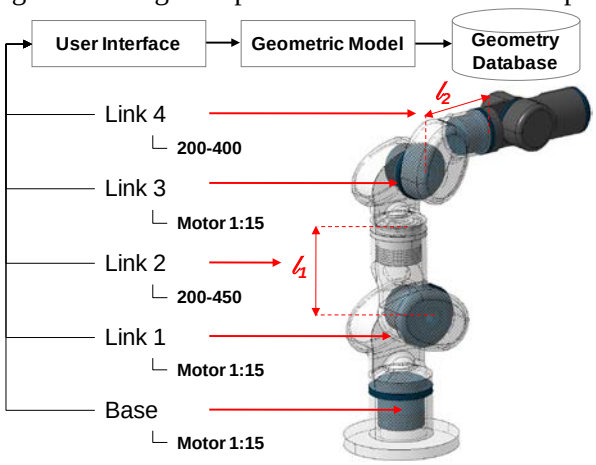

Figure 3. A geometry database is created by altering the design variables of the geometric model through a user interface.

5.3.2. Dynamic Database

Another bottleneck in the optimization framework is the dynamic simulation. A method to shorten the simulation time is thereby of importance. Storing the design configurations in a corresponding dynamic database as done for the geometric database is however not a promising approach. This is due to the kinematic and dynamic couplings between the links, both forward and backward recursively as stated in formula (2) and (3). Thereby the total number of design alternatives stored in the database would amount to tens of thousands. Consequently, in the following section another proposal is made to minimize the number of calls to the dynamic model.

5.3.3. Distributed Optimization

To further speed up optimization process, distributed optimization is utilized. The members of the robot family are thereby each distributed to a *slave PC*, as illustrated on a simplified flow chart in Figure 4. The family optimization presented in this paper consists of one *master PC* and four slaves. If the number of family members is increased, so will the number of slaves, presenting an effective means to keep the optimization time low irrespective of the number of individuals in the product family.

The optimization process starts by the master PC generating an initial population, declared as initial *Design Variables* in Figure 4. The Design Variables are utilized to calculate the commonality objective as stated in (4) and also sent to the slaves. The analyses of the robots take place in parallel, which upon completion will return the performance objective to the master, see Figure 4.

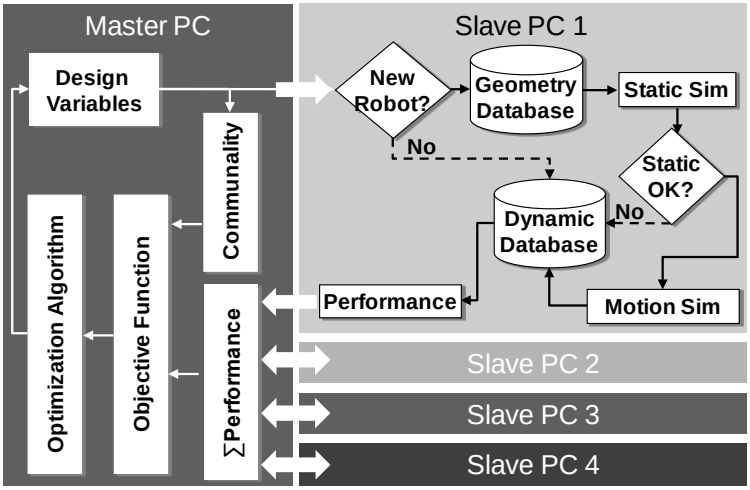

Figure 4. To speed up the evaluation process, the robot family is concurrently computed on 4 slave PCs.

For each slave PC the motion and static simulation results are stored in a dynamic database. As a result when a previously evaluated design is suggested by the optimization algorithm, the results will be retrieved from the dynamic database, thereby skipping both the static and motion simulations. If the design variables represent a new robot then the mass properties are retrieved from the geometry database and sent for static simulation. The static simulation includes a range of robot work space positions. The static simulation evaluates if the chosen motors are strong enough to withstand the gravitational forces. If the configuration does not meet the gravitational forces, then the performance objective is given a penalty value and the dynamic simulation will not be initiated, and hence the computational burden reduced. The results are stored in the dynamic database and the performance objective generated.

If the static simulation is successful then the geometrical data from the geometry database model is sent for motion simulation. The geometry database is used to parameterize the matrix and vectors in equation (1). The equation of motion for the robot is implemented in a dynamic simulation program which also includes path and trajectory planner and calculates properties such as torques, accelerations, speed and cycle-times. A set of motion cycles are simulated for each robot and the results are stored in the dynamic database and a performance objective calculated.

Although distributed computing presents faster evaluation, it is a complex procedure which needs to be properly setup otherwise it will lead to an ineffective and failed framework. The communication process involving the master and the slave is illustrated in more detail in *Figure 5*. The process starts by an initial population generated and the master sending the design variables to the slave as shown in Figure 4. The Master will then wait for a predefined time to receive a signal from the slave indicating that the design variables have been retrieved. If not, the slave PC will be terminated and another slave PC will be initiated to compute the performance objective.

The slave has a defined time at his disposal for each robot evaluation. The computation is terminated by the master as soon the time runs out. The reason for this constraint is due to some motion simulations taking several minutes to perform, whilst a preferred simulation only should take seconds. The lengthy simulation time is due to real time properties of the motion simulator, where simulation time is equal to the cycle time of each motion. Consequently, robot individuals consisting of barely strong enough motors to pass the static simulation requirement are still too weak to generate fast cycle times during the motion simulation. These individuals are weak and thus the simulation is terminated when the simulation time surpasses the time limit. A penalty is then given to the individual, which is stored in the dynamic database and the performance objective is calculated, see *Figure 5*.

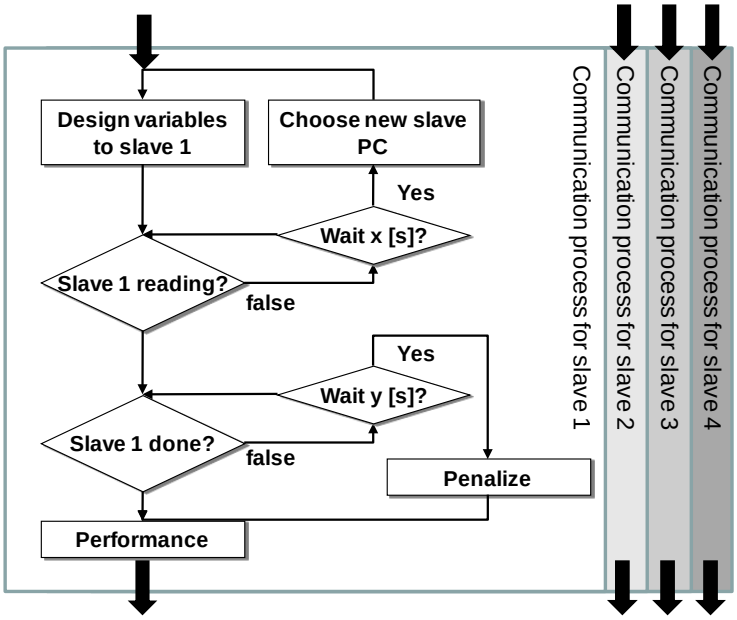

Figure 5. Communication process between the master and the slave.

5.4. Results

The outlined optimization framework is utilized to search for the Pareto frontier of the presented problem. The performance objective is to be minimized with the aim of decreasing the robot weight and cycle time, while the commonality objective is to be maximized to increase module sharing amongst the robots. In order to search for the global optimum, the following number of individuals and generations has been evaluated:

	Opt. 1	**Opt. 2**	**Opt. 3**	**Opt. 4**
Individuals	40	60	100	300
Generations	200	200	200	200

Table 2. Four sets of individuals evaluated.

Final results of the Pareto frontiers, up to the 5th rank, are visualized in Figure 6, where not surprisingly, as the number of individuals increase, the Pareto frontiers move to more optimal locations. However this movement is progressively minimized, suggesting that about 100 to 300 individuals is sufficient for finding the optimal Pareto frontier.

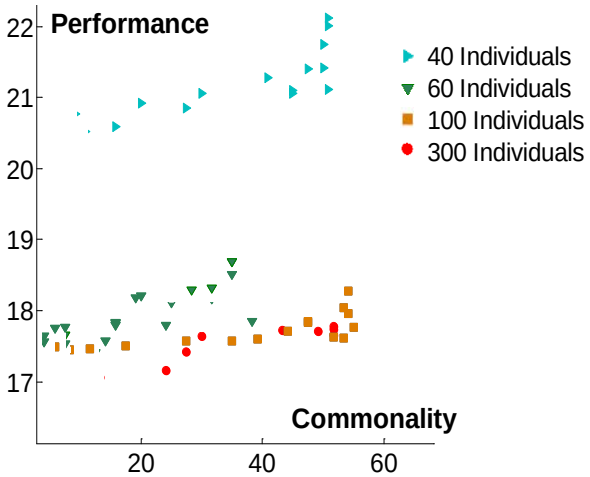

Figure 6. Pareto frontiers for 40, 60, 100 & 300 individuals.

Judging from the 1st order Pareto frontier in Figure 7, the algorithm is well suited to find solutions for both high commonality and best performance. In the robot family with best performance (1), the highest reach robot has more powerful actuators, while the smaller robots are capable in performing the pre-set trajectories with smaller actuators, hence weighing less. However the commonality level is

low. For the robot family with highest commonality (2), the actuators and arm lengths are selected in order to maximize commonality, however the overall performance is worse. The arrows on the right side of figure 7 indicate modules that are shared within the product family.

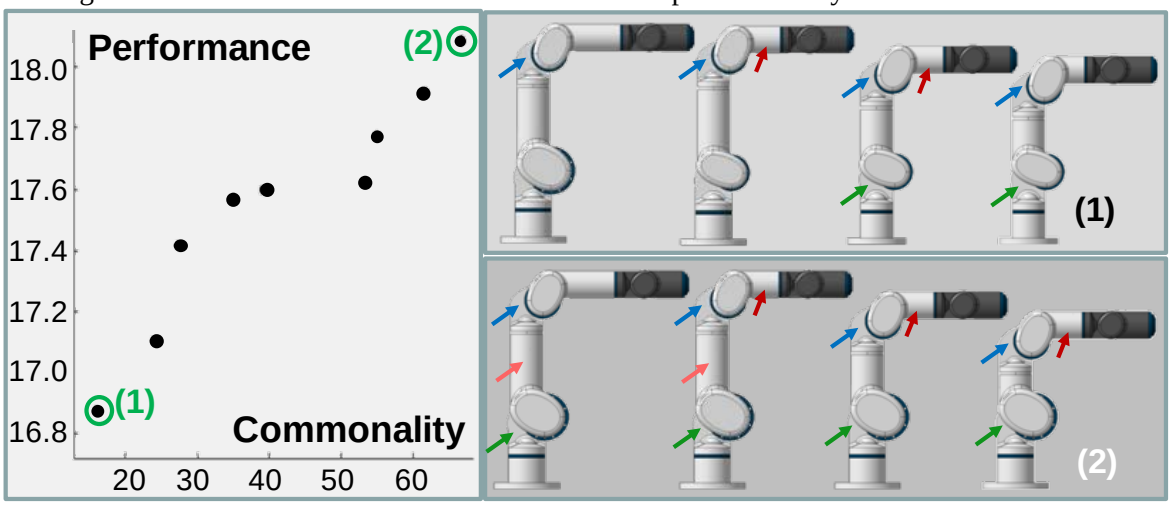

Figure 7. Pareto front of the product family, with best performance (1) and highest commonality (2). The shared modules are marked with an arrow.

6. DISCUSSION & CONCLUSION

In this paper a quantitative approach is presented where robot product family design is formulated as a formal multi-objective optimization problem. The product family design is based on tightly integrated set of high fidelity physics based models, supporting design reuse and automation. By utilizing the automated and reusable models, a MDO framework is established, facilitating automatic search for optimal robot families. An optimization case has been set up where the combination of discrete component selections invokes changes in the geometric structure, together with constraints in the dynamic simulation. The links and power train for a robot family has been optimized, and a Pareto frontier generated by applying the multi-objective genetic algorithm NSGA-II. Based on the objective functions a Pareto front is generated, presenting a range of robot families where the performance and commonality objectives have different importance. Hence, one major advantage of the presented method is that the balance between performance and commonality can be determined after generating the Pareto frontier. Therefore critical decisions can be made later in the design process, allowing engineers to gather more knowledge about the product under evaluation.

For future work, continuous variables of the actuators, e.g. maximum torque and angular velocity can be taken into consideration during the optimization. By taking the continuous variables into account the life time estimation of the drive train components can be computed and added to the performance objective. However by increasing the number of design variables, the optimization framework will have to undergo further modifications to reduce simulation time. To introduce several hierarchical layers of optimization, as well as meta-models for high fidelity models will be some of the options which should be examined further.

Another future investigation is the development of cost measurements, by estimating cost by taking commonality, component prices and life cycle in consideration.

REFERENCES

[1] Andersson, S. and Sellgren, U., "Modular product development with a focus on modelling and simulation of interfaces," Design Society – Workshop on Product Structuring, Copenhagen, January 2003.

[2] Jose A, Tollenaere M (2005) Modular and platform methods for product family design: literature analysis. J Intell Manuf 16:371–390

[3] Fellini R., Kokoloras M., Papalambros P., Perez-Duarte A., Platform Selection Under Performance Bounds in Optimal Design of Product Families, Journal of Mechanical Design, vol. 127, pp. 524-535, July, 2005.

[4] Nelson, S., Parkinson M., Papalambros P., Multicriteria Optimization in Product Platform Design, Journal of Mechanical Design, vol. 123, pp 199-204, June 2001.
[5] Erixon G., Erlandsson A., von Yxkull A., Östgren B. M., Modulindela produkten, (in Swedish), Industrilitteratur, 1994.
[6] Thevenot H., Simpson T., Commonality indices for product family design: a detailed comparison, Journal of Engineering Design, Vol. 17, No. 2, pp 99–119, 2006.
[7] Jose A, Tollenaere M (2005) Modular and platform methods for product family design: literature analysis. J Intell Manuf 16:371–390
[8] Li L., Huang G.Q., Newman S.T. A cooperative coevolutionary algorithm for design of platform-based mass customized products (2008) Journal of Intelligent Manufacturing, 19 (5), pp. 507-519.
[9] Bowcutt, K.G., "A Perspective on the Future of Aerospace Vehicle Design", AIAA 2003-6957, 12th AIAA International Space Planes and Hypersonic Systems and Technologies, Dec. 2003, Norfolk, VA, USA
[10] Giesing, J.P., Barthelemy, J-F.M., "A summary of industry MDO applications and needs", AIAA 1998-4737, AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Sept. 1998, St. Louis, MO, USA
[11] Chapman, C.B., Pinfold, M. "The application of knowledge based engineering approach to the rapid design and analysis of an automotive structure", Journal of Advances in Engineering Software, Vol. 32, Issue 12, December 2001, pp. 903-912, Elsevier
[12] Liening, A., Blount, G.N., "Influences of KBE on the aircraft brake industry", Aircraft Engineering and Aerospace Technology, Vol. 70, No. 6, 1998, pp. 439-444, MCB University Press
[13] Tarkian, M. Ölvander, J., Feng X., Pettersson M., "Design Automation of Modular Industrial Robots", ASME CIE09, San Diego, USA, Sept. 2009
[14] Hopgood, A. A., Intelligent Systems for Engineers and Scientists, Second Edition, Florida, CRC Press LLC., 2001
[15] Ledermann, C., Hanske, C., Wenzel, J., Ermanni, P., Kelm, R., "Associative parametric CAE methods in the aircraft pre-design", Journal of Aerospace Science and Technology, Vol. 9, Issue 7, October 2005, pp. 641-651, Elsevier
[16] Johanson B., Ölvander J., Pettersson M., "Component Based Modeling and Optimization for Modular Robot Design", ASME DAC'07, Las Vegas, USA, September 4-7, 2007.
[17] Petterson M., Andersson J., Krus P., "Methods for Discrete Design Optimization", proceedings of ASME DETC'05, Design Automation Conference, , Long Beach, California, USA, Sept. 2005.
[18] Sicilano B., Modeling and Control of Robot Manipulators, Springer Verlag, 2001.
[19] Spong W. Mark and Vidyasagar M Robot Dynamics and Control, John Willey & Sons Inc, pp 65-71, 1989.
[20] Goldberg, D., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley
[21] Deb K., Multi-objective Objective Optimization using Evolutionary algorithms, Wiley and Sons Ltd, 2001.
[22] Fujita K., Yoshida H., Product Variety Optimization Simultaneously Designing Module Combination and Module Attributes, Concurrent Engineering: Research and Applications, 12(2), pp105-118, 2004.
[23] Jiao J., Zhang Y., Wang Y., A Generic Genetic Algorithm for product family design, Journal of Intelligent Manufacturing, DOI:10.1007/s10845/-007-0019-7, 2007.
[24] Deb, K., Pratap. A, Agarwal, S., and Meyarivan, T. A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Transaction on Evolutionary Computation, 6(2), 181-197, 2002.