Journal of Zhejiang University-SCIENCE A (Applied Physics & Engineering)
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A novel multi-objective optimization method for the pressurized reservoir in hydraulic robotics*

Xiao-ping OUYANG, Bo-qian FAN†‡, Hua-yong YANG, Shuo DING

(*State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University, Hangzhou 310027, China*)

†E-mail: felix.zju@gmail.com

Received Jan. 24, 2016; Revision accepted Apr. 19, 2016; Crosschecked May 9, 2016

Abstract: The pressurized reservoir is a closed hydraulic tank which plays a significant role in enhancing the capabilities of hydraulic driven robotics. The spring pressurized reservoir adopted in this paper requires comprehensive performance, such as weight, size, fluid volume, and pressure, which is hard to balance. A novel interactive multi-objective optimization approach, the feasible space tightening method, is proposed, which is efficient in solving complicated engineering design problems where multiple objectives are determined by multiple design variables. This method provides sufficient information to the designer by visualizing the performance trends within the feasible space as well as its relationship with the design variables. A step towards the final solution could be made by raising the threshold on performance indicators interactively, so that the feasible space is reduced and the remaining solutions are more preferred by the designer. With the help of this new method, the preferred solution of a spring pressurized reservoir is found. Practicability and efficiency are demonstrated in the optimal design process, where the solution is determined within four rounds of interaction between the designer and the optimization program. Tests on the designed prototype show good results.

Key words: Hydraulic driven robots, Multi-objective optimal design, Interactive decision-making, Pressurized reservoir
http://dx.doi.org/10.1631/jzus.A1600034 **CLC number:** TH137.5; TP242.3

1 Introduction

Hydraulic systems have advantages of high power-weight ratio and large force output; therefore, they are widely applied to robotics as well as other locomotion machinery, such as wheel loaders and cranes. Hydraulic driven robots can be found in many applications, such as the exoskeleton robots, e.g., BLEEX (Amundson *et al.*, 2006; Zoss *et al.*, 2006) and XOS (Jacobsen, 2007), and the quadruped robots (Yang and Pan, 2015), e.g., BigDog (Raibert *et al.*, 2008) and HyQ (Semini *et al.*, 2011).

For these robots, high power density and compact size are important requirements, which can be improved by using high speed pumps (up to 10000 r/min). For a high speed pump, its inlet pressure should be maintained at the rated value (Totten and Bishop, 1999), so that fluid can be sucked into the pump steadily. Furthermore, air and contaminants should be prevented from entering the hydraulic circuit. Therefore, in the hydraulic robot with a high speed pump, the reservoir should be a close and pressurized one. The reservoir chamber is isolated from the atmosphere and accommodates changes in fluid volume. The pressurization reduces cavitation which can cause damage to pumps and valves (Vacca *et al.*, 2010). In addition, it can prevent air and contaminants being mixed into the fluid, increase the bulk modulus of the fluid, and improve the stiffness

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (No. 51275450), the Fundamental Research Funds for the Central Universities (No. 2013FZA4004), and the Science Fund for Creative Research Groups of National Natural Science Foundation of China (No. 51521064)
ORCID: Xiao-ping OUYANG, http://orcid.org/0000-0002-7763-8622; Bo-qian FAN, http://orcid.org/0000-0003-2974-0603

and response speed of the hydraulic system (Yang *et al.*, 2011).

The bootstrap is a typical closed type reservoir widely used in aircraft hydraulic systems. It is integrated with a slim actuator connected to the high pressure circuit, and self-pressurizes when the system is working. However, the bootstrap is not suitable for locomotion robotics, because its structure is complicated. Alternatively, a spring pressurized reservoir can be used in hydraulic robotics. In the spring pressurized reservoir a piston is pushed by a helical spring, so that the volume can be accommodated and the fluid is pressurized.

Components of the robotic systems are required to have a good balance of function, size, weight, strength, power ratio, etc. Usually those requirements, also known as the objectives, are in conflict with each other. The reservoir requires pressurization, installation, and stability, whilst remaining compact. Therefore, the design of a spring pressurized reservoir is a multi-objective optimization (MOO) problem.

The mathematical solution to an MOO problem is not a unique solution to a single-objective optimization problem but a set of optimal solutions named the Pareto front. A solution x_1 to an MOO problem is said to dominate another one x_2, only if no objective function value of x_1 is worse than that of x_2, and there is at least one objective value of x_1 which is better than x_2. A non-dominate solution, or Pareto optimal, is one that is not dominated by any other; the subset of all non-dominate solutions is named the Pareto front or Pareto set. Without further information, it is impossible to compare two solutions on the Pareto front. Therefore, the solution to a practical MOO design problem demands two stages, as the sketch in Fig. 1 illustrates: (1) a search for the Pareto front and (2) picking one solution on the Pareto front based on the designer's knowledge of the practical situation and its trade-offs.

Multi-objective evolutionary algorithms (MOEAs) are popular tools for solving the first stage (Fig. 1) of MOO design problems; they include the strength Pareto evolutionary algorithm (SPEA) (Zitzler and Thiele, 1998) and the non-dominated sorting genetic algorithm-II (NSGA-II) (Deb *et al.*, 2002). They are population-based, and multiple Pareto-optimal solutions can be produced after one run of MOEAs. However, there is a large gap between the evolutionary algorithms and a designer. The solution of the multi-objective optimization problem, which the MOEAs are designed to accomplish, is essentially one for finding the Pareto front, whereas the designer needs to choose one most preferred solution within the Pareto front. With a large number of Pareto-optimal solutions available, the designer may still find it hard to choose one among them. An effective method that bridges between the designer and the evolutionary algorithms, corresponding to stage (2) in Fig. 1, is greatly needed.

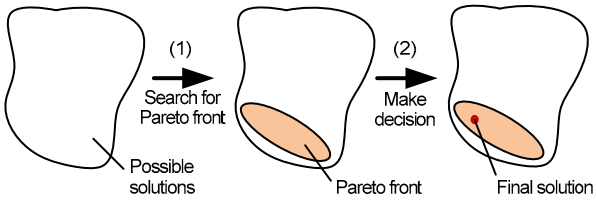

Fig. 1 Stages of MOO design problem solution

A conventional way, known as an *a priori* scheme, requires the designer preference to be provided in various ways (such as the utility function, the preference ranking, the assumed goal, and fuzzy logic) along with the problem (Coello, 2000; Rachmawati and Srinivasan, 2006). When a group of Pareto solutions are generated, the preference information is used to pick the most preferred solution, as illustrated in Fig. 2a. However, in many cases it is hard to express or model the preference information mathematically because it often involves complex trade-offs between various aspects of performance. It is a common situation where the designer knows the requirements of the problem, but has no idea about the relative importance of one objective relative to another. Moreover, in many cases the relative importance of an objective is not constant. For example, in the MOO design of a joint controller in a robot, the control frequency has a high but declining weight in the region adjacent to the required frequency, and a small weight in the higher region.

Interactive schemes, instead, require the designer to evaluate a set of Pareto solutions generated by the Pareto front search algorithm. The information

is fed back to the algorithm so that the solutions in the next interaction will be preferable. This scheme is concluded in Fig. 2b. However, many current interactive methods still depend on the preference model, which is used to identify the region of interest (Chaudhuri and Deb, 2010; Sinha *et al.*, 2014) or refine the approximation of the Pareto front (Klamroth and Miettinen, 2008). Other studies built designer preference interactively by query to the designer (Pedro and Takahashi, 2013) or comparison of pairwise solutions by the designer (Branke *et al.*, 2015; 2016); in these schemes the designer is provided with only fractional information, instead of a big picture of the optimization potential in the current situation. The visualization of Pareto-optimal solutions is also often studied in specialized topics so that the designer can make decisions based on that visual information (Kollat and Reed, 2007; Blasco *et al.*, 2008). Current graphical studies are mostly focused on objective space display of the Pareto front, and the relationship between the objective and the solution distribution is omitted.

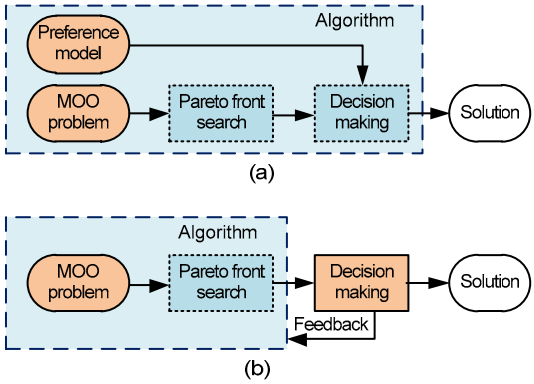

Fig. 2 The *a priori* (a) and interactive (b) methods

In this paper, a new interactive optimal design method is presented to optimize a spring pressurized reservoir which is integrated in a hydraulic robot. Instead of demanding the preference model or presenting abstract data to the designer, this method solves a complicated engineering design problem in a designer-friendly manner: the method visualizes the performance with respect to the design variables, and helps the designer by excluding the solutions that fail to meet the performance thresholds.

2 Spring pressurized reservoir

The spring pressurized reservoir is expected to work in an integrated hydraulic power unit for an exoskeleton robot, where a piston pump is driven by a combustion engine. The hydraulic circuit of the power unit is presented in Fig. 3. The pressurization is required to be no less than 0.15 MPa to prevent cavitation since the rotation speed of the engine may be as much as 10000 r/min. The volume demand is 350 ml.

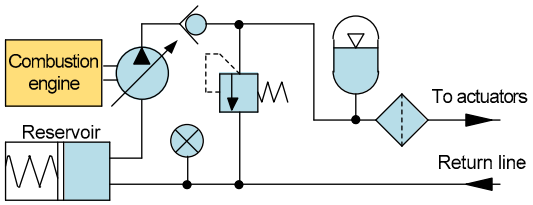

Fig. 3 Hydraulic circuit of the robotic power unit integrated with the spring pressurized reservoir

Fig. 4 demonstrates a typical spring pressurized reservoir. It mainly consists of a helical spring, a piston, two end caps (one is used to hold the helical spring and the other has inlet and outlet on it), and a housing which usually takes the form of a cylinder.

When the reservoir is in working condition, hydraulic fluid is contained in the chamber enclosed by the piston, the housing, and the end cap without air. The spring contracts or extends in response to the volume change when the fluid flows into or out of the reservoir. Meanwhile, the spring creates fluid pressure by its force.

The term design variable (DV) of the reservoir is used to address the reservoir properties which are controllable in the view of the designer. The major DVs of the reservoir consist of the DVs of the housing and the helical spring:

1. DVs of the housing: the inside diameter d_r of the cylinder and the length l_r.

2. DVs of the helical spring: the free length l_s, the coil diameter d_c, the wire diameter d_w, and the active coil number n_a. The total coil number is assumed to be $(2+n_a)$.

The wall thickness of the cylinder, and the thicknesses of the piston and the material are regarded as constants.

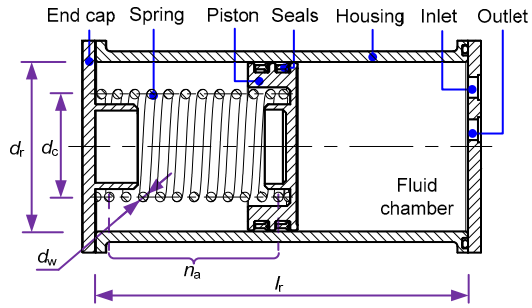

Fig. 4 Structure of a spring pressurized reservoir

The performance of the spring pressurized reservoir is indicated by its performance variables (PVs). In robotic applications, such as exoskeletons, there are many component PVs. In case of the spring pressurized reservoir, the major PVs are given as follows:

1. The initial pressure p_i is the pressure of the fluid flowing into the empty reservoir, as demonstrated in Fig. 5a. It can be calculated by Eq. (1), where l_p is the axial piston length and k_s is the stiffness of the helical spring given by Eq. (2). The initial pressure indicates the minimum pressure established by the pressurized reservoir, and is expected to be high.

$$p_i = k_s(l_s - l_r - l_p), \tag{1}$$

$$k_s = \frac{Gd_w^4}{8d_c^3 n_a}, \tag{2}$$

where G is the shear modulus of the helical spring.

2. The pressure increment p_v is the pressure increment from the initial pressure p_i to the level when the volume reaches a nominal value v_n, as Fig. 5a illustrates. Ignoring the friction between the piston and the housing, the pressure increment can be derived by Eq. (3). Ideally the pressure increment should be zero, indicating that the reservoir provides a constant tank pressure despite the change in fluid volume. However, in practice the spring force increases with the fluid volume.

$$p_v = k_s \frac{v_n}{a_r}, \quad a_r = \frac{\pi d_r^2}{4}, \tag{3}$$

where a_r is the cross-sectional area of the piston.

3. The spatial occupation v_r is the cubic space that the reservoir occupies. The nominal spatial occupation defined in Eq. (4) is used for simplicity, where the thicknesses of housing and end caps are ignored.

$$v_r = l_r d_r^2. \tag{4}$$

4. The mass of the reservoir m_r is the total mass of the reservoir, which includes the mass of the housing m_h, the piston m_p, the spring m_s, and the two end caps m_c. The mass can be calculated by

$$m_r = m_h + m_p + m_s + m_c, \tag{5}$$

and

$$m_h = \pi d_r l_r l_w \rho_h,$$
$$m_p = \pi d_r^2 l_t \rho_p / 4,$$
$$m_s = \pi d_w^2 \rho_s \sqrt{(\pi d_c n_a)^2 + l_s^2} / 4,$$
$$m_c = \pi d_r^2 l_c \rho_c / 2,$$

where l_w is the thickness of the housing and the end caps, l_t is the average thickness of the piston, and ρ denotes the material density of the parts, distinguished by the subscripts.

5. The maximum volume v_m is the maximum volume which the reservoir could hold and is assumed to be the volume when the spring is pushed to the maximum, as in the situation shown in Fig. 5b. The maximum volume can be calculated by

$$v_m = a_r[l_r - l_p - d_w(2 + n_a)]. \tag{6}$$

6. The assembling force f_a is the force on the end cap from the helical spring when the end cap is assembled on the cylinder (Fig. 5c). Normally the smaller the assembling force is, the safer and simpler the assembly process will be. If the assembling force is too large, a special instrument has to be used. The assembling force is given by

$$f_a = p_i a_r. \tag{7}$$

The design of the pressurized reservoir requires the determination of each DV, so that the PVs are

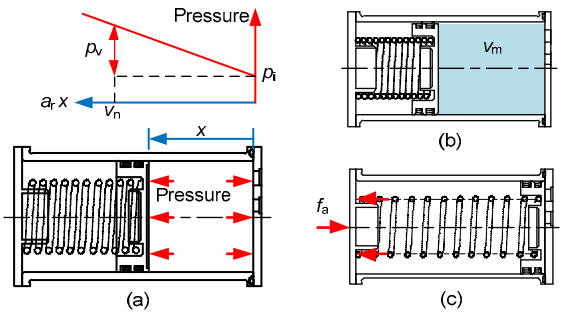

Fig. 5 Schematics of PVs: (a) p_i and p_v; (b) v_m; (c) f_a

balanced and satisfactory, according to the designer's understanding of the practical situation. The difficulties of the design problem lie in the following aspects:

1. The DVs and PVs are coupled in a complicated manner. There are many such PVs and each one is determined by multiple DVs.

2. It is hard to mathematically model the preference of each PV. As mentioned previously, the weight of a PV is not essentially constant; for example, the demand for p_i is high when p_i is less than the specified value, and much lower when p_i reaches the specified value. On the other hand, the trade-offs for a particular PV vary; for example, to reduce the mass of the reservoir from 800 g to 600 g requires much more sacrifice on other PVs than to reduce from 1200 g to 1000 g. In addition, different practical applications have unique requirements.

3. The designer needs an overview of the problem. It is common in engineering design problems that some minor requirements are hard to be quantized and integrated into the MOO models. To make a comprehensive decision, the designer needs not only an intuitive picture of the Pareto front, but also limitation and trade-off information, as well as the corresponding relation between the designs and the optimal solutions.

3 Feasible space tightening method

In this section, a novel interactive method is proposed to work as a bridge that connects the designers and the MOO solution algorithms. Generally speaking, the method involves generation of a series of 3D surface maps, named performance maps, so that the decision of the designer can be made based on the performance maps. The MOEA is adopted as the MOO solution algorithm owing to its advantages of computational efficiency as well as its reliability demonstrated in many areas of study.

3.1 Formulation of the MOO problem in engineering design

A general MOO problem in engineering design can be formulated as

$$\begin{aligned}&\boldsymbol{x}=(x_1, x_2, \cdots, x_n),\\&\boldsymbol{y}(\boldsymbol{x})=[s_1 y_1(\boldsymbol{x}), s_2 y_2(\boldsymbol{x}), \cdots, s_m y_m(\boldsymbol{x})],\\&\min \boldsymbol{y}(\boldsymbol{x}),\ \text{s.t.}\ \boldsymbol{x}\in X, g_k(\boldsymbol{x})\geq 0, k=1, 2, \cdots, r,\end{aligned} \tag{8}$$

where $\boldsymbol{x}$ is the DV vector (also known as the solution, the solution vector or the decision vector), $\boldsymbol{y}$ is the objective vector, $g_i(\boldsymbol{x})$ is the restriction function, n, m, and r are the numbers of DVs, PVs, and restrictions, respectively. The set X in Eq. (8) can be named as the search space for an MOEA, and it is the set where the DV vectors are chosen in the calculation of MOEA. The jth element of the objective vector $\boldsymbol{y}$ is the multiple of PV y_j and the direction unit variable s_j. The direction unit could be either 1 or −1, indicating that the PV y_j is desired to be as small or large as possible.

In engineering design problems, there are usually some practical constraints to the DV vector, such as that a component must be spatially smaller than its container. In this paper the DV vector that satisfies all constraints is called a valid DV vector, as opposed to an invalid one. The set of all valid DV vectors is named the valid set X^* or valid space, formulated as

$$X^*=\{\boldsymbol{x}\,|\,\forall h\in H, h(\boldsymbol{x})\geq 0\}, \tag{9}$$

where h denotes a constraint function and H is the set of all constraints.

The DV vector that satisfies every restriction $g_i(\boldsymbol{x})\geq 0$ is named a feasible DV vector, and the set of feasible DV vectors is named the feasible set or feasible space (X_f). The logical relationship of search space, valid space, and feasible space is illustrated in Fig. 6. From a mathematical point of view, the restriction $g(\boldsymbol{x})$ comes from the MOO problem, whereas

the constraint $h(\boldsymbol{x})$ describes the domain of the objective function $\boldsymbol{y}(\boldsymbol{x})$. Note that the search space is not necessarily a subset of the valid space.

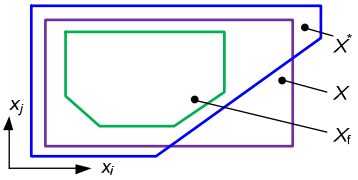

Fig. 6 Logical relationship of the search space, valid space, and feasible space (x_i and x_j are two DVs)

Another feature in engineering design is that the DVs are usually discrete, since the parameters are chosen from a priority number system. Otherwise, a dimension with many effective digits such as 6.1813 mm would bring much trouble not only in manufacturing but also in measurement and examination. In the feasible space tightening method, discrete values are assigned to the DVs rather than continuous ones. The infimum and supremum of a DV x_i after discretization are indicated as x_i^- and x_i^+, respectively; the discrete step is indicated as x_i^{h}, by which the DV is discretized into d_i steps.

3.2 Performance map

Even with the latest technology, it is not easy to visualize information in a diagram with more than three dimensions, nor is it easy to grasp the complete information in a high dimensional diagram. In addition, it is difficult to display fully a high dimensional diagram as a static image without rotating or zooming. It is even harder to demonstrate a high dimensional diagram on printed material, such as a design report. Therefore, a performance map, which can be easily displayed and understood, is adopted in the proposed method.

The performance map is a 2D surface in 3D space, formulated as Eq. (10), and used to demonstrate a PV in the 2D DV plane, or more precisely, the 2D search space named the search plane.

$$y_{k,(i,j)} = y_{k,(i,j)}(x_i, x_j), \quad (x_i, x_j) \in X_{ij}^{\mathrm{D}}, \tag{10}$$

where the superscript D in X_{ij}^{D} indicates that the search plane is discretized. The subscript $k,(i,j)$ denotes that the performance map reflects the kth PV on the x_i-x_j search plane X_{ij}^{D}.

An example of the performance map is shown in Fig. 7. The surface in the performance map is plotted with an array of discrete nodes. The location of a node P on the search space represents the discrete DV values x_i and x_j of the node. When the location of P is in the feasible space X_{ijf}, the height of P represents its PV, otherwise the height is zero (or any predefined value), as is specified by

$$\begin{aligned} & y_{k,(i,j)}^{P}(x_i^P, x_j^P) = \begin{cases} s_k \min[s_k y_k(\boldsymbol{x})], & \boldsymbol{x} \in X^{*\mathrm{D}}, \\ 0, & \boldsymbol{x} \notin X^{*\mathrm{D}}, \end{cases} \\ & \text{s.t. } \boldsymbol{x} \in X^{\mathrm{D}}, \ x_i = x_i^P, \ x_j = x_j^P, \\ & \quad g_l(\boldsymbol{x}) \geq 0, \ l = 1, 2, \cdots, r, \end{aligned} \tag{11}$$

where $X^{*\mathrm{D}}$ and X^{D} are the discretized valid space and search space, respectively. Eq. (11) could be treated as a projection of PV y_k from a subspace located by x_i and x_j of P; in the eyes of the designer, $y_{k,(i,j)}$ can be understood as how well the PV y_k would be achieved with x_i and x_j determined.

Note that from Fig. 7 the effective part of the surface is where the nodes are located in the feasible space. If the feasible space can be tightened, the number of potential solutions can be reduced, making it easier for the designer to make a decision.

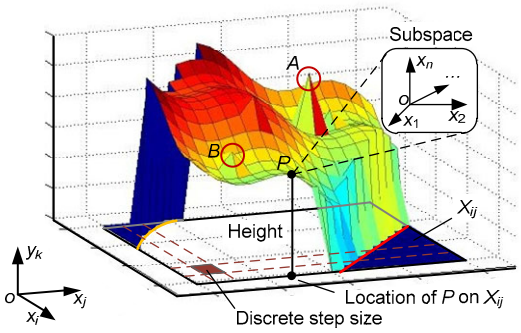

Fig. 7 Example of a performance map

On the other hand, a solution with satisfactory performance in each aspect is expected. Conversely, a solution with one high PV and another poor PV may appear on the Pareto front, but it is clearly impossible to be considered as the solution to an engineering design problem.

A measure that can simultaneously tighten the feasible space and guarantee acceptable performance is to set the threshold $y_{k\mathrm{th}}$ to performance variable y_k,

which is the key of the feasible space tightening method. Since the feasible space is bounded by the search space, the valid space, and the constraints, the only way to tighten the feasible space is to manipulate the constraints, as is given by

$$g_k(\boldsymbol{x}) = s_k[y_{k\,\mathrm{th}} - y_k(\boldsymbol{x})] \geq 0, \quad k = 1, 2, \cdots, m. \tag{12}$$

With the help of performance maps as well as the thresholds introduced, the MOO problem defined in Eq. (8) can be further divided into several subproblems by each node $P(x_i^P, x_j^P)$ on a performance map $y_{k,(i,j)}$, expressed in Eq. (13), where the first m restrictions denote the thresholds on each of the PVs, and the rest are the original r restrictions that come with the MOO problem.

$$\begin{aligned} &\min[s_k y_{k,(i,j)}(\boldsymbol{x})], \quad k = 1, 2, \cdots, m, \\ &\text{s.t.} \ \ \boldsymbol{x} \in X^{\mathrm{D}}, x_i = x_i^P, x_j = x_j^P, \\ &\quad g_l(\boldsymbol{x}) \geq 0, \quad l = 1, 2, \cdots, m, \cdots, m + r. \end{aligned} \tag{13}$$

The solutions of the MOO problem in Eq. (13) become finite after discretization of the search space. However, the number of solutions is still too large for the solutions to be fully searched when the dimension of the MOO problem is high. With the help of MOEA, it is easy to solve the optimization defined in Eq. (13), to obtain a performance map. According to the above information, a program generating the performance maps would have a structure similar to the diagram in Fig. 8. The main program works as a search engine that traverses the nodes on the search plane, as the pseudo code shows in the left column in Fig. 8. Each node is treated as an individual optimization program and is calculated by the MOEA. The calculation related to the specific problem is concentrated in the problem abstract module (PAM), with the DV generation for MOEA population as well as the calculation of PV and the constraint violation (CV) shown in the right box in Fig. 8, and called by the MOEA. The program first reads configurations, such as the search plane x_i-x_j, the search space X, and the discrete step length of each DV. Then the nodes on the search plane are enumerated, and the MOEA is run for each node. At the end of the MOEA, the most desired value of each PV is recorded. After the calculation for each node on the search plane X_{ij}, the data of each PV are derived and used to plot the m performance maps.

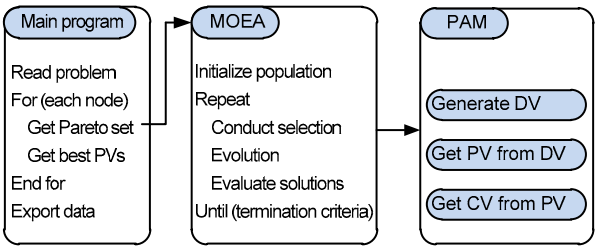

Fig. 8 Structure for a program generating the performance maps

Suppose that the generation number of MOEA in the worst condition is g, and the complexity of the MOEA on an evolutionary generation is $O[\omega(p, m)]$, where p denotes the population size of the MOEA. The computational complexity of generating the performance maps on a specific search plane is therefore $O[d_i d_j g\omega(p, m)]$ in the worst case.

3.3 Designer decision based on the performance map

On a performance map of PV y_k, the feasible plane X_{ijf} could be understood as the feasible space X_f projected on the 2D plane x_i-x_j, and the surface on the feasible space could be understood as the surface of the best PV y_k under the premise that all other PVs reached the thresholds.

At the beginning of the MOO design relatively low threshold values are chosen for each PV, since there is normally a margin for each performance criterion, and a larger feasible space can bring a better overview of the performance trends. As the designer raises the thresholds, the feasible space X_f would naturally contract in some dimensions. That is the core idea of the feasible space tightening method.

With the current thresholds, the designer knows which PV needs to improve most; after the corresponding threshold being raised, the tightening of feasible space means the decrease of candidate solutions, and is therefore a step toward the final decision. When the feasible space X_f is small enough, such that there are only 2 to 3 discrete feasible values on each DV dimension or the discrete solutions

could be enumerated, the designer will find it easy to make the decision by picking one solution from X_f.

Fig. 9 shows the flow chart of the feasible space tightening method. Although it is expected that the designer will raise the thresholds of some PVs to tighten the feasible space, there are other possible situations requiring different choices, based on the generated performance maps:

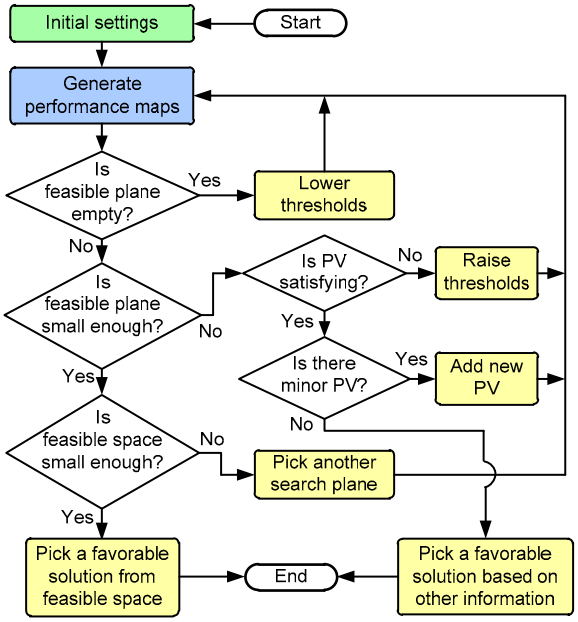

Fig. 9 Flowchart of the feasible space tightening method

1. The feasible area is empty on the search plane, which happens when the thresholds are raised too high and there is no possible solution. The designer could lower the threshold in one or several PVs, try to refine the discrete step, or adjust the search space. It may be repeated several times that the designer is not satisfied and raises the threshold, only to find the subsequent feasible space empty. In the process of trial and error, the limitation to the MOO problem can be better understood. The problem can be solved either by picking a less satisfactory but acceptable solution, or by improving the design and redefining the MOO model.

2. The designer finds that all thresholds are satisfactory, but the feasible space still contains too many solutions. The potential solutions could be unleashed by searching in the space of all DVs, it is not surprising that many more feasible solutions could be found compared with the conventional cascade design method. When the designer is content with all the PV thresholds, adding some minor PVs and corresponding thresholds is an alternative choice other than raising the threshold of the most demanding PV. Additionally, the designer could make a decision based on some requirements not related to the design method.

It can be concluded that the work flow is design thinking oriented. The designer is not required to know much about multi-objective optimization or MOEA. Instead, the designer sets thresholds to every PV as the problem demands, and picks the solution within a reduced set of feasible space by adjusting the thresholds.

3.4 Some application considerations

Some issues should be addressed when applying the feasible space tightening method.

The first one is about the interpretation of the nodes with the same location on different performance maps. For example, on the search plane x_i-x_j, there is a node $P_1^*(x_i^*, x_j^*)$ on the performance map of PV y_1 whose height is y_1^*, meanwhile there is another node $P_2^*(x_i^*, x_j^*)$ on the performance map of PV y_2 with the same location as P_1^* and height being y_2^*, as can be found in Fig. 10, which does not mean a solution x^* satisfying $y_1(x^*)=y_1^*$ and $y_2(x^*)=y_2^*$ must exist. Because y_1^* is from the projection of a node with the most desirable PV y_1 among many nodes in a subspace located by $x_i=x_i^*$ and $x_j=x_j^*$, and so is y_2^*. One can only be sure that in the subspace located by $x_i=x_i^*$ and $x_j=x_j^*$, there is a solution whose PV y_1 can be y_1^*, and there is another solution, which may be the previous one or different from it, whose PV y_2 is y_2^*.

Other issues arise in the practical implementation of the performance map-generating program based on the MOEA. It has been mentioned previously that the search space may not be a subset of the valid space. Therefore, in the MOEA initialization stage, individuals with invalid DV vectors could be generated, which need special treatment before calculation of PVs. The same problem may occur in the crossover or mutation routine, where the DV vectors of child individuals may be invalid. A simple solution to this problem is to generate a new DV vector randomly when an invalid DV vector is

detected in the PAM. Another potential issue is that the feasible space may be an empty set in the search sub-space determined by the search node, which means no valid DV vector can be found. The MOEA and PAM have to be modified to detect this situation and exit from trying to generate DV vectors.

The performance map for a real engineering design problem is usually not a smooth surface as shown in Fig. 7 or Fig. 10. There may be lots of peaks or valleys, like the node *B* in Fig. 7. If a node is too prominent than its adjacent nodes, such as the node *A* in Fig. 7, it can be referred to as an aberrant node. An aberrant node implies that the best performance of this node is probably not found. When the traversal on the search plane is completed, a quick check can be carried out to find aberrant nodes, and an additional MOEA calculation can be performed on those nodes.

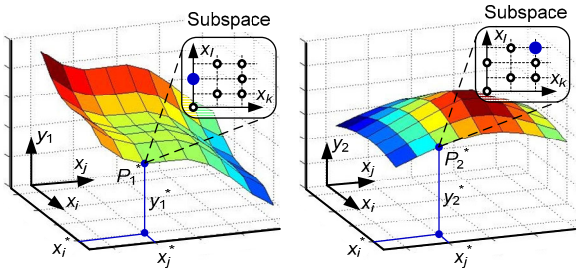

Fig. 10 Nodes with the same location on different performance maps

4 Optimization of the pressurized reservoir

The reservoir optimization problem is a 6-DV 6-PV optimal design problem with the DV vector $\boldsymbol{x}=(d_r, l_r, l_s, d_c, d_w, n_a)$ and the PV vector $\boldsymbol{y}=(p_i, p_v, v_r, m_r, v_m, f_a)$ as mentioned previously. The NSGA-II is chosen to be the MOEA in the proposed method, because of its advantages of less computational complexity which is $O(mp^2)$, and a better spread of solutions. The maximum generation number is 60 and the population number is 120. The performance map data are generated by a program written in the C language. The execution time of the program varies from 1 min to 8 min. The constant parameters for the design problem are given in Table 1, the search space and discrete step of DVs in Table 2, and the parameters for PVs in Table 3.

Table 1 Constant parameters for the design problem

Constant parameter	Value
Shear modulus of the helical spring, G (MPa)	79 000
Density of the end cap and the spring, ρ_c and ρ_s (g/cm^3)	7.85
Density of the housing and the piston, ρ_h and ρ_p (g/cm^3)	2.78
Thickness of housing wall, l_w (mm)	3.0
Thickness of end cap, l_c (mm)	3.0
Length of piston, l_p (mm)	15.0
Average thickness of piston, l_t (mm)	6.0
Nominal volume, v_n (ml)	350

Table 2 Parameters on DVs

Parameter	Value	
	Search space	Discrete step
DVs of reservoir		
Diameter, d_r (mm)	60–90	2.0
Length, l_r (mm)	140–200	5.0
DVs of spring		
Free length, l_s (mm)	180–400	10.0
Coil diameter, d_c (mm)	30–84	2.5
Wire diameter, d_w (mm)	2.5–8.5	0.5
Active number, n_a	8–25	1.0

Table 3 Parameters on PVs

Parameter	Threshold value	Direction unit
Initial pressure, p_i (MPa)	0.1	−1
Pressure increment, p_v (MPa)	0.5	1
Spatial occupation, v_r (ml)	1200	1
Total mass, m_r (g)	1000	1
Maximum volume, v_m (ml)	200	−1
Assembling force, f_a (N)	1000	1

Since there are $C(6, 2)=15$ possible combinations to choose for the search plane, the total number of performance maps is $6C(6, 2)=90$. Only the ones on the prime DVs and which reveal information on performance trends are used by the designer. In this case three search planes are used, namely the housing exterior plane (d_r-l_r), the spring exterior plane (l_s-d_c), and the spring intrinsic parameter plane (d_w-n_a). The first two describe the shape of the housing and spring, respectively, and the third one determines the intrinsic parameters of the spring.

In Fig. 11, five performance maps on the d_r-l_r search plane demonstrate the best possible

performance for different housing sizes. In the worst case, the minimum p_i is 0.15 MPa, the maximum p_v is 0.1 MPa, m_r is 839 g, and f_a is 652 N. There is a notable margin between the worst case p_v, f_a and the corresponding thresholds, indicating the optimization potential of those PVs. It can be seen from Fig. 11 that the PV of the initial pressure and assembly force are insensitive to the housing length. When the performance map of the assembly force is viewed on the l_r axis direction, as Fig. 11f shows, the surface in the feasible space is overlapped and appears to be a curve. The pressure increment can be smaller for a short and wide housing; meanwhile, the volume and mass can be smaller for a short and slim one.

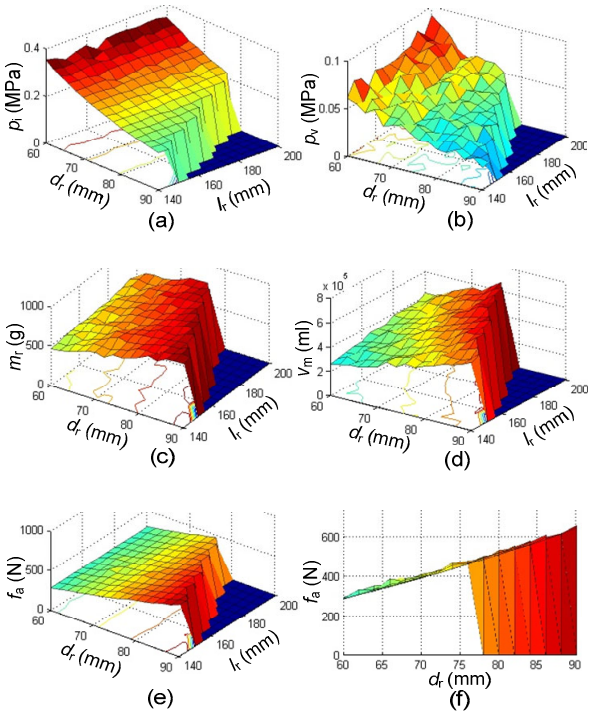

Fig. 11 Performance maps on d_r-l_r plane
(a) Initial pressure; (b) Pressure variation; (c) Total mass; (d) Maximum volume; (e) Assemble force; (f) Assemble force (side view)

Another four performance maps are shown in Fig. 12 with respect to the helical spring parameter l_s and d_c. In Fig. 12b, the pressure increment falls from a high value as the spring free length increases from the boundary of the search space. Besides, the performance trends indicate that a long but slim helical spring is preferable (marked in Fig. 12a). However, such a helical compression spring tends to buckle and may not be stable (Shigley *et al*., 1989). Therefore, a new PV, named the dimension ratio r_s, given in Eq. (14) could be adopted as a qualification of the helical spring design, and should be a small value.

$$r_s = l_s / d_c. \tag{14}$$

With the current performance index, the threshold of mass is set to 800 g, maximum volume 350 ml, assembly force 800 N, and dimension ratio 7.0, because those PVs are currently the main concern. In addition, the threshold of initial pressure is set to 0.15 MPa and the pressure increment to 0.2 MPa. After raising the thresholds, typical performance maps are obtained as shown in Fig. 13.

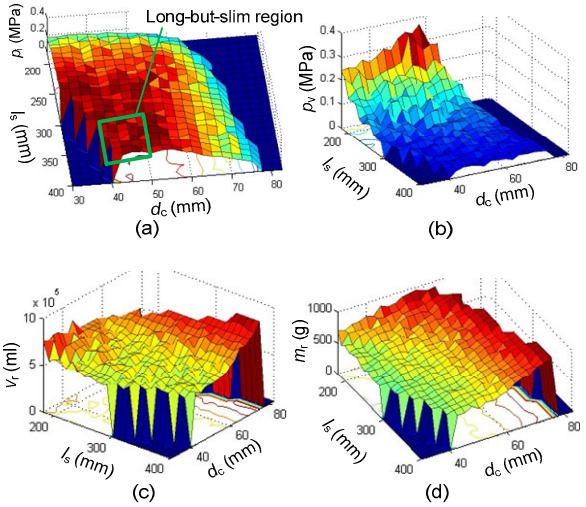

Fig. 12 Performance maps on l_s-d_c plane
(a) Initial pressure; (b) Pressure variation; (c) Spatial occupation; (d) Total mass

It is obvious that the feasible plane of the performance maps in Figs. 13a–13c is much smaller than those in the corresponding maps in Figs. 11 and 12, which is within the expectations of the feasible space tightening method. The feasible plane in Fig. 13d is at one boundary of the search plane, which implies that the search space is possibly inappropriate. Meanwhile, plate mounting is a good choice for the reservoir because of its compactness. However, a slim but long housing, as in the region marked in Fig. 13a, is not easily mounted firmly and reliably. This is because the mounting surface is on

the end cap, and a slim but long housing will bring a large overturning force due to gravity or vibration. A simple way to avoid this situation is to add another PV, the dimension ratio of housing r_h, and the corresponding threshold to the design problem. The dimension ratio of housing is defined in Eq. (15), which is similar to Eq. (14). The PV r_h is desired to be a small value.

$$r_h = l_r / d_r. \tag{15}$$

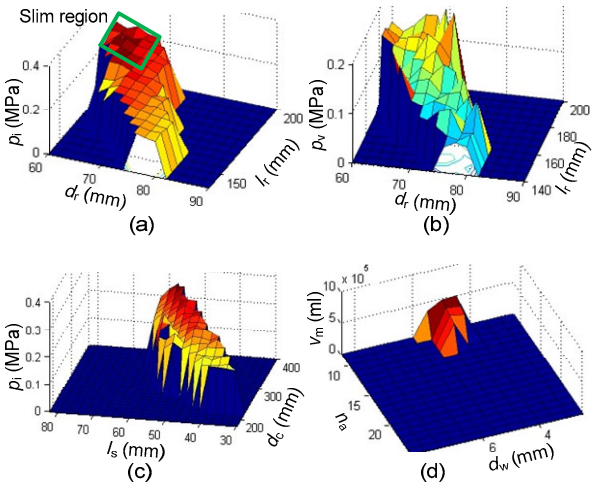

Fig. 13 Performance maps after raising of thresholds
(a) Initial pressure on d_r-l_r; (b) Pressure variation on d_r-l_r; (c) Initial pressure on l_s-d_c; (d) Maximum volume on d_w-n_a

The threshold of r_h is set to be 2.4, the search space for n_a is modified to be 6.0–12.0 with search step of 0.5. A new group of performance maps are generated, which are shown in Fig. 14, with threshold adjustment: $v_r \leq 1000$ ml and $v_m \geq 450$ ml.

The increment of the initial pressure in Fig. 14a is small, which means the initial pressure is now insensitive to the DVs on this search plane. Other PVs, such as m_r, v_m, and f_a, show similar trends. The pressure increment shows a clear trend: it decreases as spring length increases. On the d_r-l_r plane, the feasible region is reduced, as shown in Fig. 14c due to limitation of the housing dimension ratio. The feasible region on the d_w-n_a plane is also small.

The shape of the housing can be determined as d_r=80 mm and l_r=170 mm, so that the reservoir can have a maximum volume up to 598 ml; meanwhile, the other PVs are above thresholds. After this only four DVs of the helical spring remain. New performance maps can be generated, given in Fig. 15, with d_r and l_r treated as constant values.

The feasible nodes are marked by white circles in Fig. 15. Even though the thresholds are unchanged, the feasible region is becoming smaller, because the subspace of each node is greatly reduced with d_r and l_r determined. Next, the shape of the helical spring is determined as d_c=48 mm and l_s=310 mm, to achieve small pressure increments. After this, only one node on the d_w-n_a search plane remains, which is d_w= 4.5 mm, n_a=6. Thus, the preliminary design of the spring pressurized reservoir is complete. The parameters are all determined, and the designer can move on to tasks, such as verification and detailed design. The parameters of the design are summarized in Table 4, as well as the predicted performance. The design derived by the method is believed to accurately reflect the demands of the application, because each time the PV maps are presented to the designer, the PVs that cause most concern are improved by the designer.

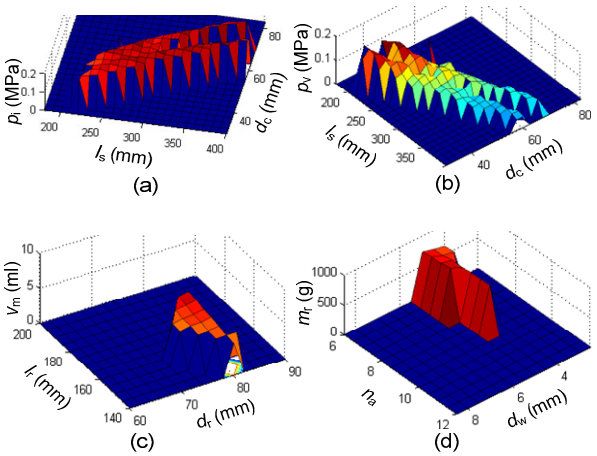

Fig. 14 Performance maps on the second tightening
(a) Initial pressure on l_s-d_c; (b) Pressure variation on l_s-d_c; (c) Maximum volume on d_r-l_r; (d) Reservoir mass on d_w-n_a

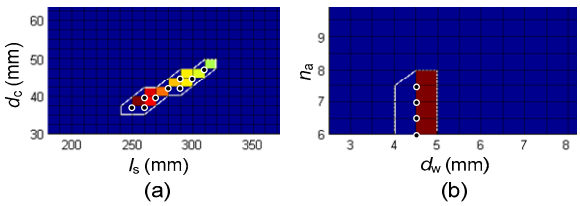

Fig. 15 Feasible space with d_r and l_r determined
(a) Feasible region on l_s-d_c; (b) Feasible region on d_w-n_a

Table 4 Parameters and performance of the design

Parameter					
d_r (mm)	l_r (mm)	l_s (mm)	d_c (mm)	d_w (mm)	n_a
80	170	310	48	4.5	6
Major performance indicator					
p_i (MPa)	p_v (MPa)	v_m (ml)	m_r (g)	v_r (ml)	f_a (N)
0.152	0.08	1088	796	598	763

5 Test of the reservoir prototype

A spring pressurized reservoir prototype was manufactured according to the above optimization results, and was integrated into the hydraulic power unit for exoskeleton robots, as shown in Fig. 16.

Fig. 17 shows the dynamic behavior of the reservoir, when the hydraulic system of the exoskeleton robot is working. The movement of one actuator in the robot is given in Fig. 17a to represent the working cycle, with a frequency of 0.5 Hz; the system pressure is also presented. The dynamic behavior of the reservoir, shown in Fig. 17b, is characterized by the volume and pressure in the reservoir (i.e., the pressurization). The volume decrement is observed when the fluid fills the accumulator (Fig. 3) as the system pressure rises.

It can be seen that the variation of pressurization is not in proportion to the variation of volume, as is the static characteristic of the spring pressurized reservoir. This is the result of the friction between the piston and the housing as well as the inertia of the piston. Nevertheless, the reservoir is able to maintain the required pressure.

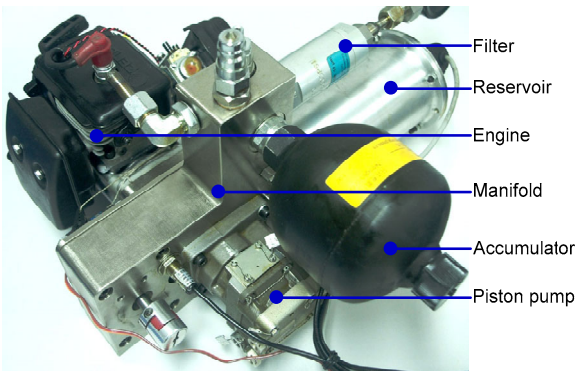

Fig. 16 Hydraulic power unit with spring pressurized reservoir

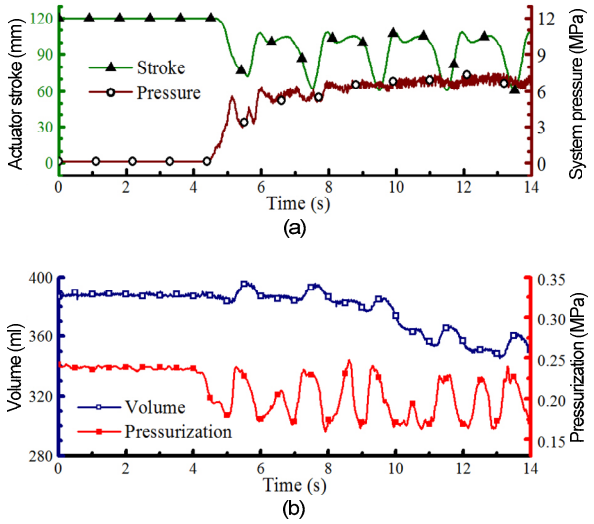

Fig. 17 Operating characteristics of the spring pressurized reservoir

(a) Movement of one actuator; (b) Dynamic behavior of the reservoir

6 Conclusions

A new multi-objective optimal design method named the feasible space tightening method is presented to solve sophisticated engineering design problems with multiple design variables and multiple performance variables (objectives), where the utility function, cost, or relative importance of each objective is hard to model.

The method visualizes the performance with respect to the design variables by 3D surfaces entitled performance maps, with which the designer can reduce the feasible space based on the thresholds determined interactively, and the design progress can become much easier. The method has advantages in that:

1. There is a good distribution of Pareto solutions. Every node on a performance map can be viewed as an individual optimization problem and is solved by MOEA in the subspace of the node.

2. There is ability to solve high dimensional engineering design problems. The high dimensional PV-DV space is projected on to several performance maps and displayed. Every PV would be acceptable

or satisfying, hence the overall performance of the design can be guaranteed.

3. The visualization of PV-DV space is intuitive. With the help of performance maps, the designer can learn information on the feasible space and the trends of PVs on the chosen search space.

4. The flow is designer-friendly. The work flow is design-thinking oriented. The designer is not required to know much about multi-objective optimization or MOEA. Instead, the designer sets thresholds to each PV as the problem demands, and chooses the solution within a reduced set of feasible spaces by adjusting the thresholds.

With this method, a spring pressurized reservoir is parametrically developed within four rounds of interaction between the designer and the optimization program. In each round of interaction, the most crucial objective is addressed by the designer; meanwhile, less but more preferable solutions are generated by the program. Test results of the reservoir prototype show the expected good performance.

References

Amundson, K., Raade, J., Harding, N., *et al.*, 2006. Development of hybrid hydraulic–electric power units for field and service robots. *Advanced Robotics*, **20**(9):1015-1034. http://dx.doi.org/10.1163/156855306778394058

Blasco, X., Herrero, J., Sanchis, J., *et al.*, 2008. A new graphical visualization of *n*-dimensional Pareto front for decision-making in multiobjective optimization. *Information Sciences*, **178**(20):3908-3924. http://dx.doi.org/10.1016/j.ins.2008.06.010

Branke, J., Greco, S., Slowinski, R., *et al.*, 2015. Learning value functions in interactive evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, **19**(1):88-102. http://dx.doi.org/10.1109/TEVC.2014.2303783

Branke, J., Corrente, S., Greco, S., *et al*., 2016. Using Choquet integral as preference model in interactive evolutionary multiobjective optimization. *European Journal of Operational Research*, **250**(3):884-901. http://dx.doi.org/10.1016/j.ejor.2015.10.027

Chaudhuri, S., Deb, K., 2010. An interactive evolutionary multi-objective optimization and decision making procedure. *Applied Soft Computing*, **10**(2):496-511. http://dx.doi.org/10.1016/j.asoc.2009.08.019

Coello, C.A.C., 2000. Handling preferences in evolutionary multiobjective optimization: a survey. Proceedings of the Congress on Evolutionary Computation, La Jolla, CA, USA, p.30-37. http://dx.doi.org/10.1109/CEC.2000.870272

Deb, K., Pratap, A., Agarwal, S., *et al.*, 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, **6**(2):182-197. http://dx.doi.org/10.1109/4235.996017

Jacobsen, S., 2007. On the development of XOS, a powerful exoskeletal robot. IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA.

Klamroth, K., Miettinen, K., 2008. Integrating approximation and interactive decision making in multicriteria optimization. *Operations Research*, **56**(1):222-234. http://dx.doi.org/10.1287/opre.1070.0425

Kollat, J.B., Reed, P., 2007. A framework for visually interactive decision-making and design using evolutionary multi-objective optimization (VIDEO). *Environmental Modelling & Software*, **22**(12):1691-1704. http://dx.doi.org/10.1016/j.envsoft.2007.02.001

Pedro, L.R., Takahashi, R.H., 2013. Decision-maker preference modeling in interactive multiobjective optimization. *In*: Evolutionary Multi-criterion Optimization, Springer Berlin Heidelberg, p.811-824. http://dx.doi.org/10.1007/978-3-642-37140-0_60

Rachmawati, L., Srinivasan, D., 2006. Preference incorporation in multi-objective evolutionary algorithms: a survey. IEEE Congress on Evolutionary Computation, Vancouver, BC, USA, p.962-968. http://dx.doi.org/10.1109/CEC.2006.1688414

Raibert, M., Blankespoor, K., Nelson, G., *et al.*, 2008. Bigdog, the rough-terrain quadruped robot. Proceedings of the 17th World Congress, COEX, South Korea, p.10822-10825.

Semini, C., Tsagarakis, N.G., Guglielmino, E., *et al.*, 2011. Design of HyQ–a hydraulically and electrically actuated quadruped robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, **225**(6):831-849. http://dx.doi.org/10.1177/0959651811402275

Shigley, J.E., Mischke, C.R., Budynas, R.G., *et al.*, 1989. Mechanical Engineering Design. McGraw-Hill, New York, USA.

Sinha, A., Korhonen, P., Wallenius, J., *et al.*, 2014. An interactive evolutionary multi-objective optimization algorithm with a limited number of decision maker calls. *European Journal of Operational Research*, **233**(3):674-688. http://dx.doi.org/10.1016/j.ejor.2013.08.046

Totten, G.E., Bishop, R., 1999. The Hydraulic Pump Inlet Condition: Impact on Hydraulic Pump Cavitation Potential. SAE Technical Paper No. 1999-01-1877. http://dx.doi.org/10.4271/1999-01-1877

Vacca, A., Klop, R., Ivantysynova, M., 2010. A numerical approach for the evaluation of the effects of air release and vapour cavitation on effective flow rate of axial piston machines. *International Journal of Fluid Power*, **11**(1):33-45. http://dx.doi.org/10.1080/14399776.2010.10780996

Yang, H.Y., Pan, M., 2015. Engineering research in fluid

power: a review. *Journal of Zhejiang University-SCIENCE A (Applied Physics & Engineering)*, **16**(6): 427-442.
http://dx.doi.org/10.1631/jzus.A1500042

Yang, H.Y., Feng, B., Gong, G.F., 2011. Measurement of effective fluid bulk modulus in hydraulic system. *Journal of Dynamic Systems, Measurement, and Control*, **133**(6):061021.
http://dx.doi.org/10.1115/1.4004783

Zitzler, E., Thiele, L., 1998. Multiobjective optimization using evolutionary algorithms—a comparative case study. *In*: Parallel Problem Solving from Nature—PPSN V. Springer Berlin Heidelberg, p.292-301.
http://dx.doi.org/10.1007/BFb0056872

Zoss, A.B., Kazerooni, H., Chu, A., 2006. Biomechanical design of the Berkeley lower extremity exoskeleton (BLEEX). *IEEE/ASME Transactions on Mechatronics*, **11**(2):128-138.
http://dx.doi.org/10.1109/TMECH.2006.871087

中文概要

题　目：液压机器人增压油箱的多目标优化方法研究

目　的：增压油箱是提高液压机器人动力源功率密度的一个关键元件。高集成度的增压油箱设计涉及 6 个设计变量和 6 个性能指标，必须采用合适的方法进行多目标优化。

创新点：1. 提出一种在设计变量平面上投影性能曲面的多目标优化方法，通过设定性能阈值缩小可行解范围并获得决策；2. 将增压油箱应用于液压机器人，提高液压机器人的功率密度和性能。

方　法：1. 采用活塞-弹簧增压的原理来实现机器人液压系统增压，分析增压油箱的容量、质量和增压压力等性能，确定增压油箱设计为多目标优化问题。2. 通过在设计变量平面上的投影曲面，分析增压油箱性能指标与设计变量之间的关系；将目标函数阈值引入设计限制条件，通过控制待优化的指标缩小可行域，获得油箱设计的最终解。3. 按优化设计参数加工油箱样机，并在液压机器人动力源上进行测试。

结　论：1. 增压油箱优化结果表明本文提出的设计方法可帮助设计者获得所需的最优解；2. 增压油箱样机的应用测试结果表明所研制的增压油箱在液压机器人系统中运行可靠。

关键词：液压机器人；多目标优化；交互式决策；增压油箱