2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Layout algorithm of searching point for a CFD optimization problem

Yoshifumi Kuriyama, Ken'ichi Yano and Mamoru Watanabe

Abstract— **Optimization with the aid of computational fluid dynamics (CFD) simulators has recently attracted attention in various fields. Optimization is often impeded by the increased computing time used by a large number of localized solutions, which complicate the calculation by consuming vast amounts of memory for generation and distribution of solutions. Therefore, researchers are actively seeking methods which would allow them to accelerate the process and make the analytical time highly effective. In this paper, we propose an algorithm that obtains the optimal solution by considering the layout of the search points for high-speed convergence performance. We also verify the effectiveness of the algorithm after applying it to an actual problem.**

I. INTRODUCTION

Numerical simulators for fluid analysis based on computational fluid dynamics (CFD) focus on analyzing the behavior and the thermal hydraulics of a fluid flowing around an object. CFD is a technique that considers the Navier-Stokes equations and the energy conservation laws and uses the mass conservation method. With the ongoing increase of computational power and the concurrent decrease in the price of personal computers, CFD simulators have become a useful and practical tool for analyzing real-world problems [1]. Furthermore, CFD is currently used not only for analysis of the behavior of fluids, but also for optimization of the shape and the flow of fluids for improved quality and performance of various products.

Nevertheless, optimization with a CFD simulator for improved quality or performance is still facing a number of problems. For example, the solution space formed by the optimization solutions obtained by using a CFD simulator becomes a multimodal space with a large number of local minimums due to the effect of underflow errors or the loss of significant digits. Furthermore, optimizations for practical use become more complex since such applications require more variables and constraints.

A meta-heuristic algorithm is a heuristic method commonly employed for performing an efficient search for a complex solution space [2]. Furthermore, the genetic algorithm (GA) is generally used as the algorithm with the greatest versatility[3]∼[6]. However, in cases such as analyzing a problem that contains a large number of local solutions, the general GA is highly unlikely to escape from a local solution, and thus it is difficult to derive the global optimized solution. Naturally, this problem can be solved by increasing the number of population members, the number of generations and the mutation evolution. On the other hand, the computation of one condition currently requires a few minutes, and the entire optimization requires hundreds of repeated computations. Thus, a considerable amount of time is required in order to perform an optimization by using a CFD simulator.

The aim of this study is to design a search point algorithm that requires a smaller population to find a solution and that can be applied to the production of high-quality products. Thus, we propose a multi-subcenters solution search algorithm for computing the optimal plunger velocity in die-casting. The effectiveness of the proposed system is shown through both simulations and experiments.

II. MULTI-SUBCENTERS SOLUTION SEARCH ALGORITHM

In the case of optimization with few search points, the distribution of search points is important for the derivation of the global optimal solution. Conventionally, search points are selected at random. However, the distribution of such search points is uneven, as shown in Fig. 1, where the shaded areas in the figure contain no search points. Such areas without search points make it difficult to find the global optimal solution with GA, leading to greater computational cost.

Thus, the search points should be distributed uniformly. On the other hand, in order to improve the convergence performance, the search points should be densely located in areas that are expected to contain the global optimal solution. Therefore, to satisfy these criteria, we developed the

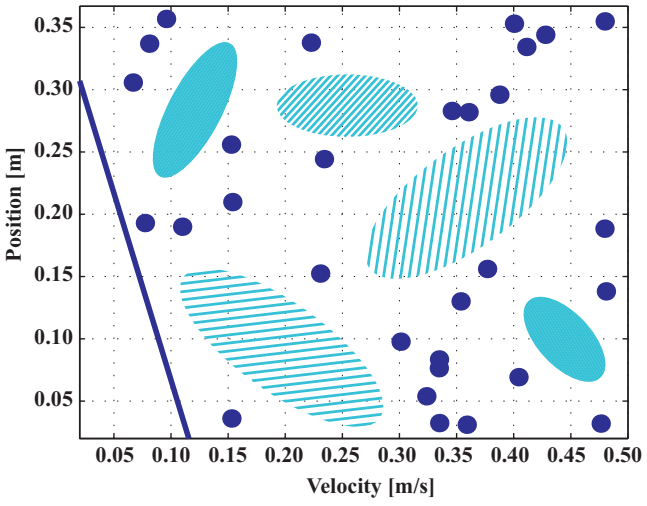

Fig. 1. Setting the distribution of search points by using random generation.

Y. Kuriyama is with Faculty of Engineering, Gifu University, 1-1 Yanagido, Gifu 501-1193, Japan n3812202@edu.gifu-u.ac.jp
K. Yano is with Faculty of Engineering, Mie University, 1577 Kurimamachiya-cho, Tsu , Mie 514-8507, Japan
M. Watanabe is with Terrabyte Co.,Ltd., NOV bld.2F, 3-10-7 Yushima, Bunkyo, Tokyo 113-0034, Japan

multi-subcenters solution search algorithm, which features distribution control [7], convergence control and cluster analysis capabilities, where the distribution control algorithm is responsible for the distribution of search points and the convergence control algorithm regulates their density.

A. Distribution algorithm

Fig. 2 shows the basic concept of the distribution algorithm. The search points are moved by repulsive forces, which are represented as extending circles in the figure.

In this distribution algorithm, each search point is first placed in a unique circle, which expands as the calculation progresses, as described by (1), where $R(\lambda)$ is the radius of the circle, R_{add} is the expansion factor added to the radius, λ is the number of cycles and m_q is the motion vector of the search point.

$$R(\lambda+1) = \begin{cases} \forall R(\lambda) + R_{\text{add}}, & \text{if } \forall m_q = 0 \\ \forall R(\lambda), & \text{otherwise} \end{cases} \tag{1}$$

When a circle touches another circle or the boundary, repulsive forces arise and spread the circles apart. The search points are moved by the repulsive forces in accordance with (2), (3) and (4), where F_{rep} is the repulsive force, q is a search point, p is another search point, F_{lim} is the repulsive forces reacting from the boundary, $B_{\min}$ and $B_{\max}$ are respectively the upper and lower boundaries, q_k is the k-th column in vector q. Also, n is the number of variables, R_p is the radius of vector p and R_q is the radius of vector q.

$$m_q = \begin{cases} (q-p)/F_{\text{rep}}^2, & \text{if } \|q-p\| \le R_q + R_p \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$F_{\text{rep}} = (\|q-p\|)^{-1}, \ (q \ne p) \tag{3}$$

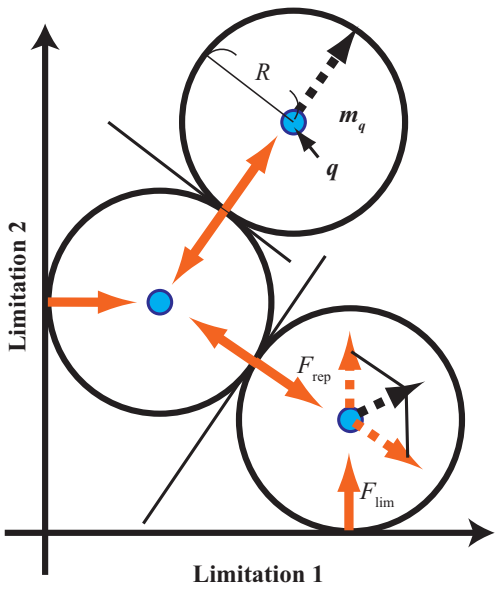

Fig. 2. Basic concept of distribution algorithm.

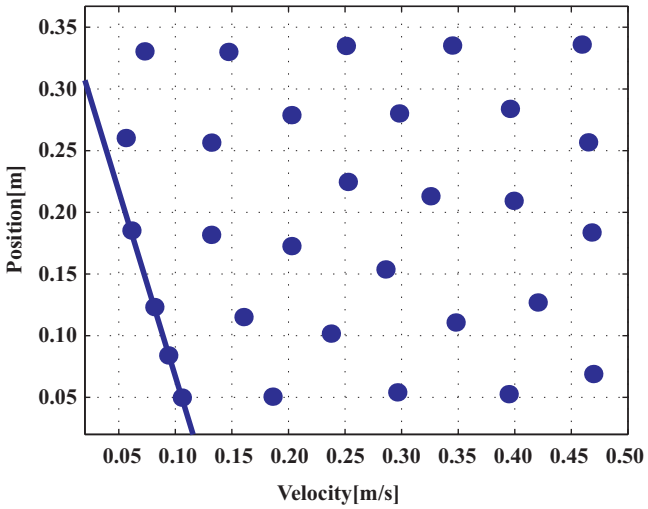

Fig. 3. Setting the distribution of search points by using the developed distribution algorithm.

$$m_q = \begin{cases} -F_{\text{lim}}, & \text{if } q_k - R \le B_{\min,k} \\ F_{\text{lim}}, & \text{if } q_k + R \ge B_{\max,k} \\ 0, & \text{otherwise} \end{cases} \quad (1 \le k \le n) \tag{4}$$

The circles continue to expand until the movement of the search points stops. When this occurs, the search points should be distributed uniformly. Equation (5) is applied as an additional constraint on the movement of the search points.

$$q(\lambda+1) = \begin{cases} q(\lambda), & \forall f(q(\lambda) + \sum m_q) \notin L \\ q(\lambda) + \sum m_q, & \text{otherwise} \end{cases} \tag{5}$$

Here, f is a constraint function defined by the user, and L is the constraint. The distribution of search points which results form using the distribution algorithm is shown in Fig. 3, where the edges of the figure indicate the upper and lower boundaries, and the inner solid diagonal line indicates the additional constraint condition. In contrast to Fig. 1, the search points are distributed uniformly. Thus, the distribution algorithm can ensure sufficient separation between the search points.

B. Convergence algorithm

After the distribution algorithm, the convergence algorithm is executed, in which the search points converge in accordance with neighboring evaluated points. Fig. 4 shows the basic concept of the convergence algorithm, where the crosses in the figure indicate evaluated points. If there are evaluated points in the vicinity of a search point, the latter is moved as shown in the figure. Here, R_{max} is the effective range of a neighboring evaluated point, which is determined by the maximum value of R used in the distribution algorithm.

Fig. 5 presents the basic concept of the movement of search points.

The movement of search points is governed by (6)∼(11). The search points of q search for nearby evaluated points present inside the circle. The center of gravity coordinates g of the selected evaluated points are calculated by using (6).

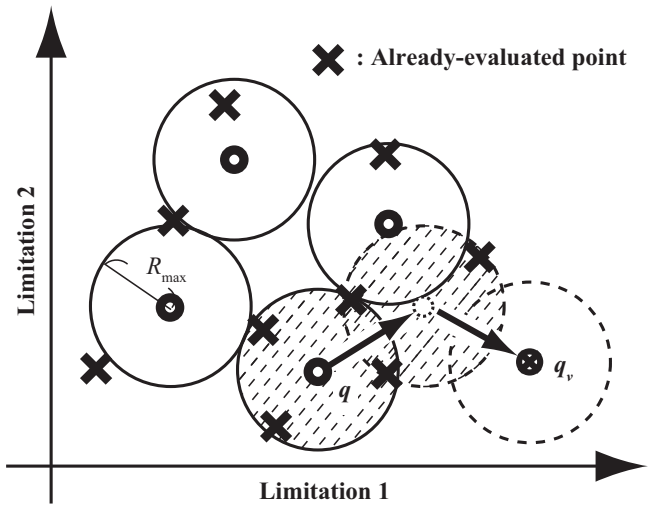

Fig. 4. Basic concept of the convergence algorithm.

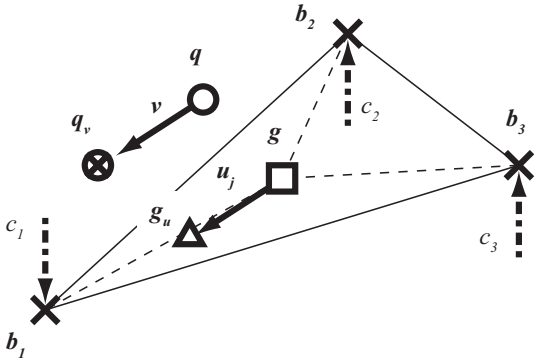

Fig. 5. Basic concept of the movement of search points.

Furthermore, the center of gravity coordinates g_u, where the evaluated value of c_i is considered as the load, are calculated by using (7). Then, the motion vector u_j, which is given by (8) is calculated from (6) and (7), where b_i is the position vector of the evaluated point.

$$g = \left(\sum_{i=1}^{n+1} b_i \right) (n+1)^{-1} \tag{6}$$

$$g_u = \left(\sum_{i=1}^{n+1} b_i c_i^{-1} \right) \left(\sum_{i=1}^{n+1} c_i^{-1} \right)^{-1} \tag{7}$$

$$u_j = g_u - g \tag{8}$$

The movement of a search point is decided by the motion vector u_j, as described by (9), (10) and (11).

$$v_q = \begin{cases} 0 & (d = 0 \text{ or } \forall f(q+v_q) \notin L) \\ \left(\sum_{j=1}^{d} u_j \right) d^{-1} & \text{otherwize} \end{cases} \tag{9}$$

$$q_v = q + v_q \tag{10}$$

Here, q_v is the moved search point, v_q is the motion vector of the search point and d is the number of combined evaluated points as given by (11). If the number of evaluated points a is less than $n+1$, then d=0 is applied, as shown in (11).

$$d = \begin{cases} {}_aC_{n+1} & (a \geq n+1) \\ 0 & (a < n+1) \end{cases} \tag{11}$$

These calculations are continued until the movement stops. The shifted points become the analyzed points of the next generation.

Furthermore, an attractive force algorithm is employed in order to enhance the overconcentration performance of the distribution of search points. Fig. 6 shows the concept of the attractive force algorithm. The search points are moved in accordance with (9). In case a search point enters the range of an attractive force while being moved, the attractive force influences the movement of the search point, and its motion vector rotates to point towards the attractive force point.

The attractive force point x_g is the evaluated point satisfying (12), where X is the set of evaluated points, w_R is the weighting factor given by the user, $w_R R_{\max}$ is the effective range of the attractive force, k is the number of the current generation, which is multiplied by $w_R R_{\max}$ in order to aggregate the search points inside the space more efficiently. Also, b_g is the evaluated point satisfying (13), where c_g is the cost of b_g, $\bar{B}$ is the average cost of all evaluated points and b_a is the threshold value, which is given by the user.

$$x_g = \min \left\{ f(x_g) | x_g \subseteq X : \| x_g - b_g \| \leq w_R R_{\max} k \right\} \tag{12}$$

$$b_g : c_g / \bar{B} \leq b_a \tag{13}$$

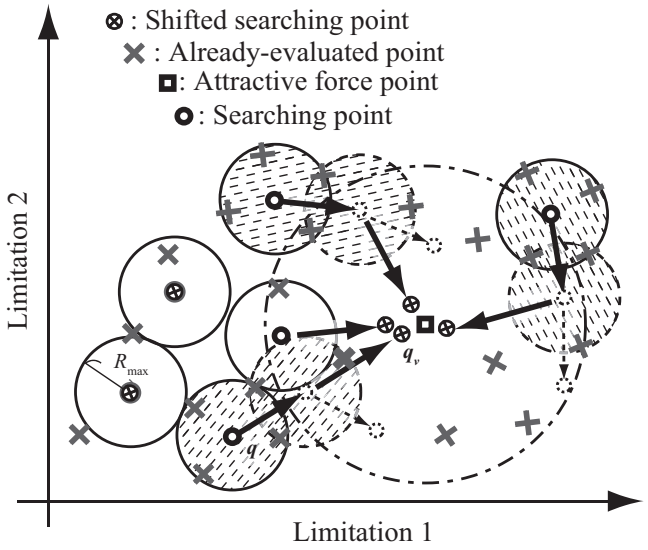

Fig. 6. Attractive force algorithm

III. Application of the Schwefel Function

The Schwefel function is used in a performance test aiming to determine the effectiveness of the proposed method. The analysis range was taken between $0 \leq x_j \leq 500 (j = 1 \sim n)$, where x_j is an integer, the minimum point is $x^* =$

$[421,\cdots,421]$, and n is the number of variables. In this performance test, we compare the convergence generation to find the minimum point.

Table I shows the parameters of GA, and Table II shows the parameters of the proposed method, as set by trial and error in order to obtain high convergence performance.

TABLE I
PARAMETERS FOR GA

Number of populations	30
Number of elite preservations	2
Mutation fraction	0.03
Crossover fraction	0.80

TABLE II
PARAMETERS FOR MSSA

Number of populations	30
R_{add}	0.05
F_{lim}	0.1
w_R	0.1
b_a	0.3

Fig. 7 shows the generations at which convergence is obtained in the case of $n = 5$, where the maximum number of generations is 250. Regarding the median value of the generation at which convergence is obtained, the GA requires 143 generations to find the minimum point, whereas the proposed method requires 64 generations. This result indicates that the convergence performance in this case is improved by 55 percent.

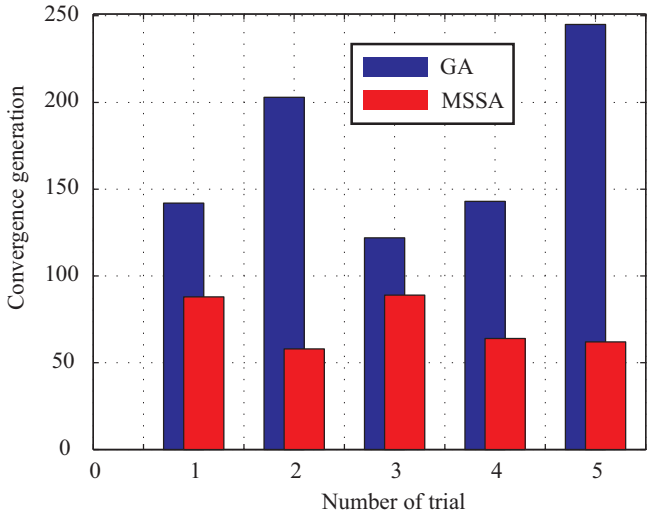

Fig. 7. Generations at which convergence of the optimization using GA and MSSA ($n = 5$) is obtained

IV. APPLICATION TO DIE-CASTING PROCESS

The effectiveness of the proposed method is tested through application to die-casting, and a comparison with GA is performed.

At actual casting plants, the multistep velocity can be controlled, and the velocity pattern, which consists of five phases, is derived from past studies ([8], [9]). The plunger velocity is expressed as shown in Fig. 8. In this study, the velocity is set to v_1,v_2,v_3, and the acceleration distance is set to x_1,x_2, where x_3 is the filling position, which takes a constant value of 0.367 m due to the constraints of the plant.

TABLE III
FLUID PROPERTIES OF ADC12

Density of fluid [kg/ m^3]	2700
Viscosity of fluid [Pa·s]	0.0030
Specific heat [J/(kg· K)]	1100
Thermal conductivity [W/(m· K)]	100.5
Initial temperature [K]	653.15

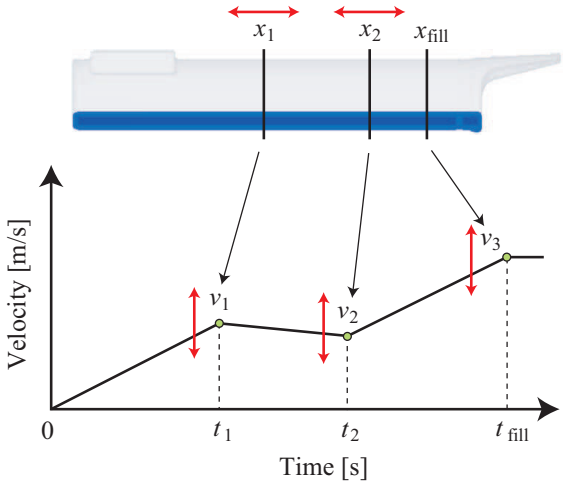

Fig. 8. Die-casting simulation model using 5 variables.

In this study, the plunger tip is flat, and hot-work die steel (SKD61) is used for the die, the sleeve and the plunger. An aluminum alloy (ADC12) is assumed as the molten metal. Table III shows the fluid properties of ADC12. The die temperature during pouring is set in the range between 110 and 150°C (steady state), and the temperature of the molten metal in the melting furnace is set to a value between 660 and 680°C. Yushiro AZ7150W is used as a parting agent.

Fig. 9 shows an overview of the mesh setting, and Table IV lists the parameters for the mesh setting.

TABLE IV
MESH PARAMETERS.

	Cell size	Number of cells
X-direction	0.004	20
Y-direction	0.002∼0.006	132
Z-direction	0.0022∼0.0035	29
Total number of cells		76500

As seen in Fig. 9, the sleeve is symmetrical about the X axis. Thus, the area targeted in the analysis is regarded as having only one side in order to reduce the analysis time to only about ten minutes. Table IV shows the minimum settings necessary for performing calculations quickly and

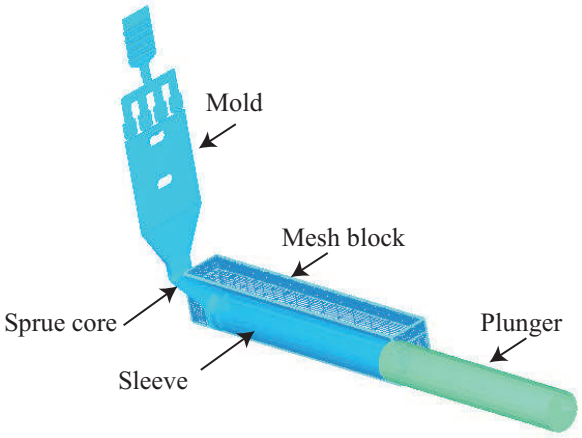

Fig. 9. Mesh settings for CFD simulation.

accurately, and the mesh parameter is set such that a rough mesh is used around the start point of the sleeve because the velocity is low and the fluid is stable in that section. On the other hand, a fine mesh is used around the end point of the sleeve because the turbulence at the early stage of filling caused by collision with the sprue core in that section.

The optimization problem is defined with a cost function equivalent to the sum of the weighted quantity of air entrainment and the weighted filling time, as shown in (14),

$$\begin{aligned} \text{minimize :J} \quad &= \quad w_a A(v_i, x_i) + w_t t_3 \\ &\quad + K_p \end{aligned} \tag{14}$$

$$\begin{aligned} \text{subject to :} \quad & 0.02 \leq v_i \leq 0.60 \\ & 0.02 \leq x_i \leq 0.36 \\ & 0 \leq t_i \leq 2.0 \\ & A_{\text{shut}} \leq 3.0 \end{aligned} \tag{15}$$

Here, A is the function of air entrainment, t_3 is the filling time, x_i is the acceleration distance, and $w_a = 1.5$ and $w_t = 1.0$ are weighting factors. K_p is a penalty applied each time the conditions shown in (15) are not satisfied. A penalty $K_p = 10^8$, which is sufficiently large to avoid the penalty conditions, is added to satisfy the criteria. Also, A_{shut} is the volume of trapped air, not including the air surrounding the sleeve wall, the plunger and the molten metal, when the plunger injection is switched from low speed to high speed. A_{shut} is defined as shown in Fig. 10.

Three parameters are introduced to calculate the amount of air entrapment.

- D_1: Volume/opening column of fluid in the Y cross-section.
- D_2: Threshold of the amount of air entrapment.
- D_3: Calculation time step.

We use fluid analysis software to perform calculations at each time interval specified by D_3 to output the cell column fraction and the fraction of fluid in order to determine the filling for the cross-section of the sleeve. We calculate the space volume at the back, where the fraction of fluid is less than $D_1 \times 100\%$ for the cross-section, and we also define

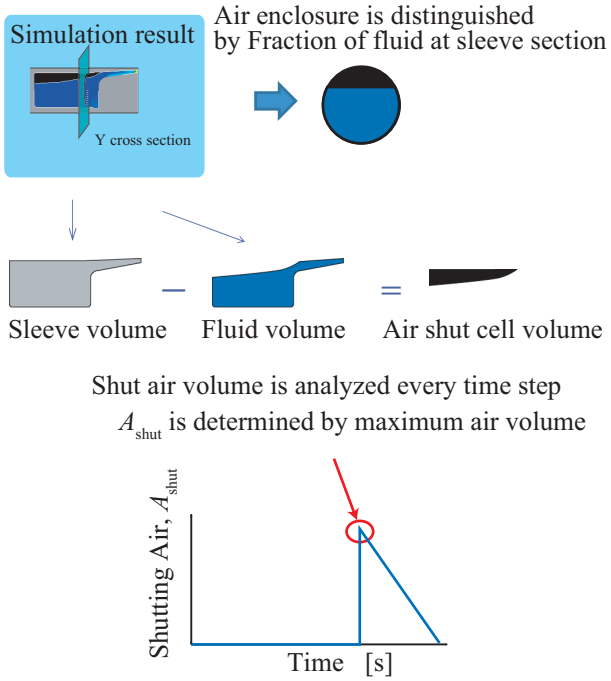

Fig. 10. Time progression for air trapped inside the sleeve.

the maximum space volume as the amount of air entrapment A_{shut} m^3. If a plunger velocity input which minimizes air entrapment can be designed by using this simulator, favorable results can be expected for experiments targeting actual production.

V. Optimization of the Plunger Velocity

The performance of the proposed method is validated for the case of plunger velocity controlled, where the initial population is the same for each algorithm, allowing us to perform the calculations under the same conditions.

The results of the calculations performed for both the proposed method and GA are shown in Table V.

TABLE V

Performance Comparison of First Optimization Results.

Parameter	Proposed method	GA
Cost function J	1.789	1.794
Air entrainment A	0.193	0.203
Finish time [s]	1.50	1.489
Generation at which convergence occurs	39	39

VI. Experimental Results

Next, we present the results of experiments performed at an actual die-casting plant. In the experiments, the optimal velocity inputs calculated with both the proposed method and GA were used. The optimized parameters calculated by using the two methods are listed in Table VI. A blister test was carried out to investigate the quantity of entrapped air. This was performed by heating the specimen in a furnace, which increased the pressure of the entrapped air and formed

blisters on the surface. Fig. 11 shows the total area of the surface of the test piece obtained from each calculated value for the optimal velocity, where air bubbles formed as a result of the blister test. Here, the solid lines indicate air bubble areas larger than 1×10^{-6} m^2, broken lines indicate air bubble areas smaller than 1×10^{-6} m^2, and the conventional input used in [8] is indicated as a comparison. From the figure, we can see that the optimal velocity calculated with the proposed method resulted in the formation of fewer air bubbles as compared with GA. Moreover, the results in Table VI show that the performance of the proposed method was higher in comparison to GA even when the cost function was similar. The experimental results demonstrate that optimization with the proposed method corresponds accurately with the favorable results obtained in the experiment.

TABLE VI
FIRST RESULTS OF OPTIMIZATION EXPERIMENT.

Proposed method	Velocity [m/s]	Position [m]
i=1	0.41	0.242
i=2	0.34	0.340
i=3	0.59	0.367
Cost function 1.789		
Total area of the air bubble : 0.779×10^{-6} [m^2]		
GA	Velocity [m/s]	Position [m]
i=1	0.44	0.279
i=2	0.22	0.290
i=3	0.60	0.367
Cost function : 1.794		
Total area of the air bubble : 0.913×10^{-6} [m^2]		

VII. CONCLUSION

The purpose of this study was to design a solution search algorithm that requires smaller populations to find the optimal solution corresponding to the production of a high-quality product. Specifically, we proposed the MSSA(Multi-subcenters Solution Search Algorithm) for computing the optimal plunger velocity for die-casting. The results of several experiments showed that in comparison with GA, the proposed method was capable of obtaining a superior optimization of the plunger velocity in die-casting, resulting in the entrapment of less air. Moreover, the results obtained by using the proposed method were more favorable than those obtained with GA even when the cost function was similar. The experimentation results demonstrated that optimization with the proposed method corresponded directly and accurately to superior die-casting production.

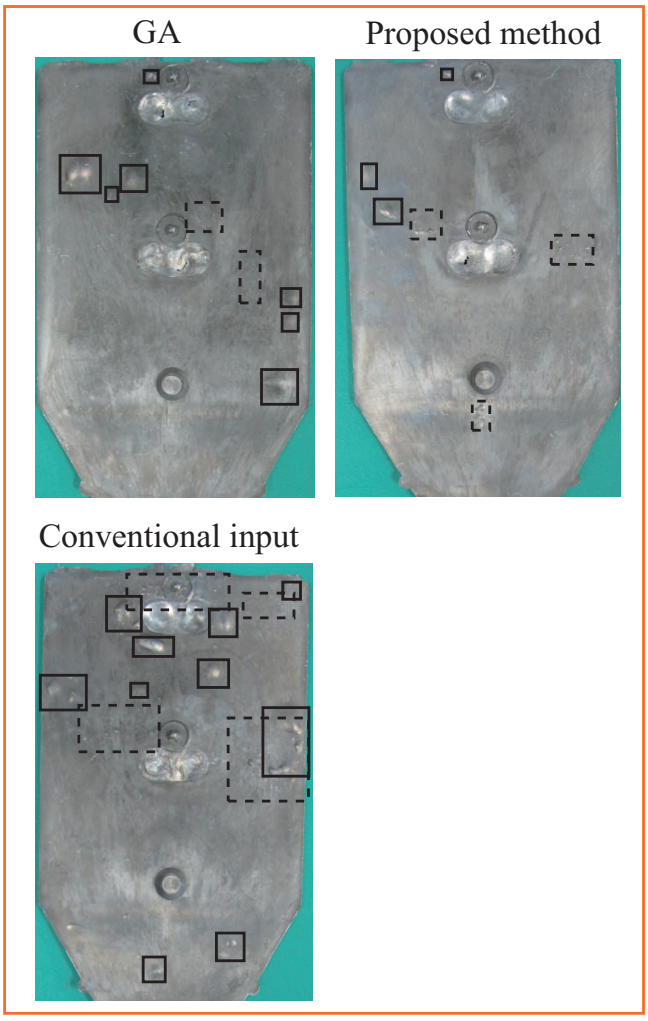

Fig. 11. Results of optimization experiment performed by using the proposed method.

REFERENCES

[1] P. Stefano P. Carlo M. Martin, "Integrating Multibody Simulation and CFD : toward Complex Multidisciplinary Design Optimization," JSME international journal. Ser. B, Fluids and thermal engineering, vol.48, no.2, 2005, pp. 224-228

[2] P. Moscato, "gentle introduction to memetic algorithm," Handbook of Metaheuristics, 2003, pp. 105-144

[3] T. Bäck, H. P. Schwefel, "An Overview of Evolutionary Algorithm for Parameter Optimization," Evolutionary Computation, vol.1, no.1, 1993, pp. 1-23

[4] K.Ikeda and S.Kobayashi. GA based on the UV-structure Hypothesis and Its Application to JSP 6th Parallel Problem Solving from Nature, pages 273–282, September 2000.

[5] I. Ono, and S. Kobayashi, "A Real-coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover," Proc. 7th Int'l Conf. on Genetic Algorithms, 1997, pp. 246-253

[6] K. Ikeda, S. Kobayashi, "GA based on the UV-structure Hypothesis and Its Application to JSP," 6th Parallel Problem Solving from Nature, 2000, pp. 273-282

[7] S.Martinez, J.Cortes and F.Bullo. Motion Coordination with Distributed Information.IEEE Control System Magazine, volume 27, pages 75–87, 27, 2007.

[8] Ken'ichi Yano, Kotaro Hiramitsu, Yoshifumi Kuriyama, Seishi Nishido, "Optimum Velocity Control of Die Casting Plunger Accounting for Air Entrapment and Shutting," International Journal of Automation Technology, vol.2, no.4, 2008, pp. 259-265

[9] Y.Kuriyama, S.Hayashi, K.Yano and M.Watanabe,Solution search algorithm for a CFD optimization problem with multimodal solution space.Proc. of IEEE Conference on Decision and Control, pages 5556–5561, December 2009.