소뇌모델 선형조합 신경망의 구조 및 학습기능 연구(II)
— 학습 시뮬레이션 및 응용 —

On Learning and Structure of Cerebellum Model Linear Associator Network (II)

—Learing Simulation & Engineering Application—

황 헌*, 백 풍 기**
H. Hwang, P. K. Baek

적 요

연구 I에서 수행한 소뇌모델 선형조합 신경망(CMLAN)의 분석 결과와 제안된 능률적 학습 알고리즘들에 의거하여 이차원 비선형 함수치의 출력 모의시험과 팔의 형태에 따라 두개의 목적치를 갖는 2 자유도 머니퓨레이터의 동작지령 산출 모의시험을 행하였다. 특히 2 자유도 머니퓨레이터의 경우, 작업공간에 적절한 입력네트의 변수를 선정하고, 하나의 입력공간을 공유하는 두개의 세부 소뇌모델 선형조합 신경망을 서로 연결하는 구조로써 팔의 형태와 목적 지점에 따라 네트를 선정하는 구조를 갖도록 하였다.

제안한 학습 알고리즘의 성능 및 CMLAN의 학습에 따른 효과를 학습이득에 따라 컴퓨터로 모의시험하였으며 그 결과를 분석하였다. 잘 알려진 신경망인 BACK-PROPAGATION 다층(Multi-Layer) 신경망과 함수연결 신경망(Functional Link Net)을 이용한 모의시험 결과를 비교 분석하였다. CMLAN의 학습 능률성은 학습에 소요되는 컴퓨터의 cpu시간과 학습 중의시스템의 최대 편차와 RMS 편차의 변이도 및 최종 시스템 수렴치로서 나타내었다.

1. INTRODUCTION

CMLAN(Cerebellum Model Linear Associator Net) is a schematical approximate modeling of the memory driven information processing of the cerebellum. CMLAN has a simple structured processing nature of generating output in response to any continuous or discrete state input. Because of this fact, CMLAN has been applied to various applications as a control substitute or a reference input generator and visual image processor and revealed its usefulness at various levels in control problems. So far their applications were based on CMAC(Cerebella Model Arithmetic or Articulated Controller) under the maximum error correction algortihm which was proposed and analyzed by Albus[2, 3)

Albus[1, 5] implemented CMAC module to control a seven degree of freedom master-slave arm. CMAC was used in a closed loop control system to make the slave arm follow a specific trajectory. The trajectory was trained by the feedback from the movement of the master arm. CMAC was used to control a two degree of freedom biped walking de-

* 成均館 大學校 農業機械工學科
** 建國大學校 農業機械工學科

vice by Camana[6]. Three degree of freedom planar manipulator was trained to follow a trajectory to avoid collisions under a fixed static obstacle by Rajadhyaksha[7]. Reference trajectories were generated manully using a stylus and trained iteratively.

An adaptive hierarchical model for two dimensional computer vision was simulated using CMAC by Manglevdakar[8]. Three hierarchical levels based on the segmentation were implemented to analyze the image formed from a digitizer tablet.

Miller[9] applied CMAC to position the manipulator with visual feedback by training on-line observations of the input-output relationship of the system being controlled. Learning parameters were chosen on an ad hoc basis. CMAC based control of a two degree of freedom articulated robot arm was performed for the simulated repetitive and non-repetitive movements by Miller et al[10]. The control activity of CMAC in conjunction with a fixed-gain linear feedback controller was tested. A simulated learning of a manipulator was performed every one cycle of trajectory trace using on-line information based on the Albus's maximum error correction training.

Unfortunately, those previous CMAC applications have been done without fully analyzing net intself and feasible learning algorithms in detail. As mentioned in Part I, since CMAC works similar to the linear asociator net which is one of well known models of Neural Net, instead of CMAC, CMLAN is more proper name of the net considering its characteristics. In Part I, the detailed anaysis on the structure and function of CMLAN has been done and three basic learning rules such as a batch type accumulated sequential error learning, on-line type direct sequential error learning, and the uniformly distributed random error learning have been proposed and verified to be far more efficient over the conventional maximum error learning.

Various artificial neuro-nets have been devloped via modeling the structural and functional characteristics of human brain in the hope of achieving highly adaptive and sophisticated information processing with high rate of computing[11-12]. It should be noted that researches on how to apply those developed neuro-nets as a tool properly to the various engineering problems are hot issues as well. Every enginerring application requires the proper management of system parameters and the modification of the net structure such as connection of nodes and layers including net and system control parameters.

In this paper, using the previously mentioned learning algorithms in Part I, a nonlinear function geneartor and a motion generator for a two d.o.f. manipulator were simulated. Trained results were compared using the rms and maximum errors over the sampled input nodes for the equivalent learning period. The rms and maximum errors over the extended input nodes were also compared to show the generalizing effect of each learning. The uniform quantizing scheme mentioned earlier was applied.

2. Nonlinear Function Generator

A nonlinear function, $P=sin(x)\ sin(y)$ (fig. 1) was trained with input ranges of $0 \leq x \leq 360$ and $o \leq y \leq 180$(deg) using the batch type SEC(G=0.5), on-line type SEC(G=0.8), MEC(G=0.8), and REC(G=0.4) learnings. The interval of sampled input nodes was 15 degrees resulting 325 pairs of inputs. The offset of CMLAN input space was 1 degree and K was equal to 30. The learning gain of the batch type SEC was selected from the scheme presented in part I.

The measured cpu times were 3.83 seconds and 4.44 seconds per learning epoch for batch and on-line type SEC learnings respectively. It took 2.62 seconds per epoch and 0.012 seconds per each random learning for MEC and REC respectively. The rms and maximum errors were compared over the sampled input nodes and the extended input nodes of 3 degree interval (including untrained nodes) and those are shown in fig. 2 and fig. 3. Note that the trained performance of each learning has been scaled based on the cup time elapsed. Total learned cpu time can be obtained by simply multiplying cpu time per epoch for the SEC and MEC learning and cpu time per each learning for the REC learnings. Remember that the batch type SEC learning and MEC learning execute a process of learning actually once per epoch.

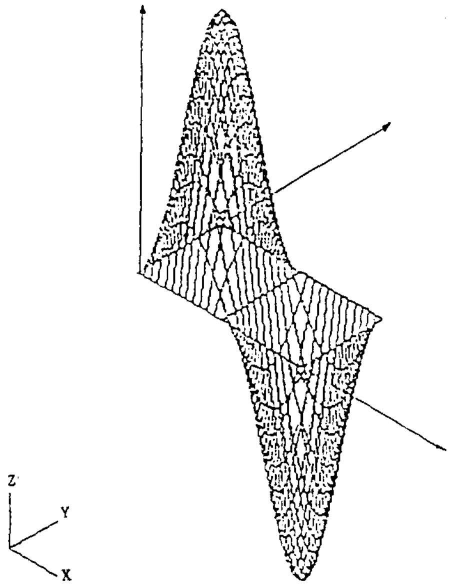
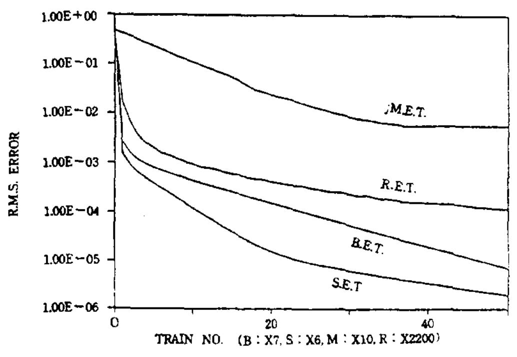
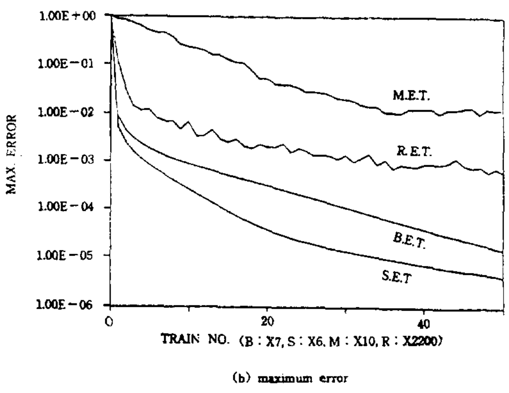


Fig. 1. A nonlinear function to be trained $P=sin(x)\ sin(y)$ with $o<x<360$ and $o<y<180$ (deg).

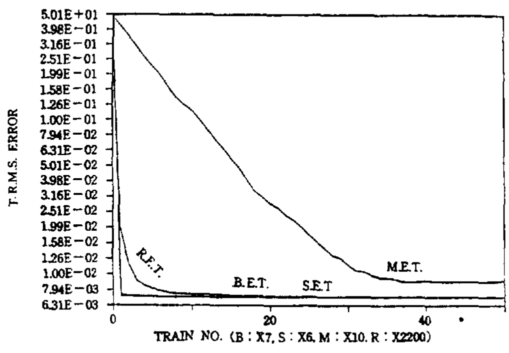


(a) rms error

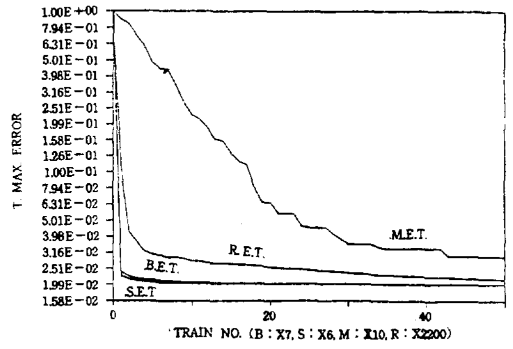


(b) maximum error

Fig. 2. The performance using various learning over sampled input nodes : (a) rms error, (b) maximum error (Sampled interval=15 deg, Batch SEC : G=0.5, On-line SEC : G =0.8, MEC : G=0.8, REC : G=0.4).



(a) rms error



(b) maximum error

Fig. 3. The performance using various learnings over extended input nodes : (a) rms error, (b) maximum error (Sampled interval=3 deg, Batch SEC : G=0.5, On-line SEC : G=0.8, MEC : G=0.8, REC : G=0.4).

The SEC and REC learnings showed the excellent error fitting on the sampled input nodes. However, they show rather poor results on the overall node space sampled by 3 degree. This reveals their linear interpolating effect on the intermediate untrained notes, which are connected as nonlinear surface.

Maximum and rms error have almost same trends as the number of training epoch increases. The rms error oscillates in the REC and MEC learnings as learning number increases although it is not shown clearly in fig. 2. The REC learning shows a rather large oscillating behavior in maximum error because of its randomly selected correction without considering the slope of overall error surface. The MEC learning shows the worst learned rms error performance and the oscillating behavior in maximum error even it corrects the selected node of maximum error at each epoch.

By comparing CMLAN results with those of the learning simulations for the similar two dimensional Gaussian surface performed by Pao(13) using the functional link net and the generalized back propagation delta rule(11-12), the power of CMLAN is noticeable. The functional link net does a similar mapping of the input state vector as CMLAN does by utilizing the functional expansion and outer product pattern enhancement, which allows to use a flat net architecture for learning a desired function. Instead, CMLAN automatically generates an input pattern enhancement by its unique structured mapping.

3. JOINT MOVEMENT OF TWO DOF MANIPULATOR

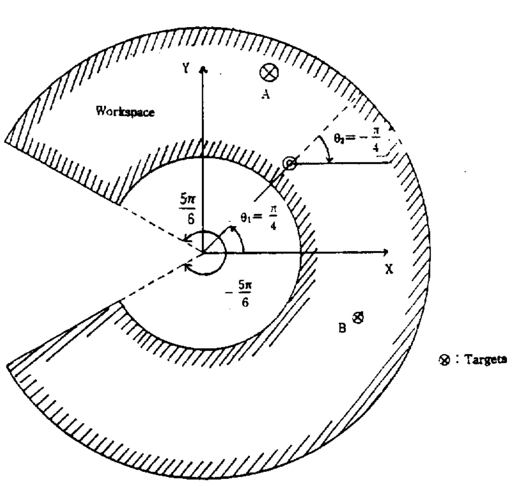
Two d.o.f. planar manipulator with a length of arm one 400mm and arm two 300mm was trained to move from the specified initial arm state to the target point in the work space by training the required joint movements directly. The initial arm configuration was set such that $\theta_1 = \pi/4$ rad and $\theta_2 = -\pi/4$ rad. Since two different arm configurations exist for every target point, the final arm configuration and the entire joint movements were determined from the initial joint states and the location of the target shown in fig. 4.

The CMLAN input space is the work space of the manipulator and was defined by two inputs using polar variables, $r = (300, 700)$ in *mm* unit and $\phi = (-150, 150)$ in degree unit. Fig. 5 shows configurations of two separated desired delta joint movements from the initial states, which CMLAN should learn, and aspects of discontinuity caused by two different desired arm patterns.

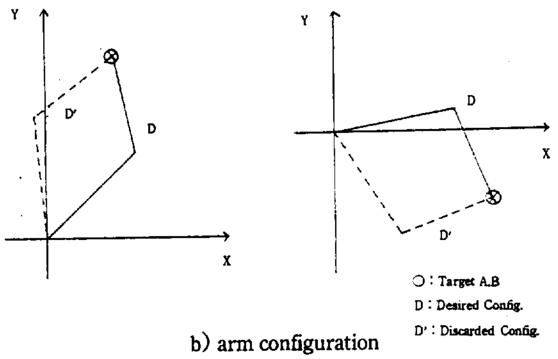
Two separate sets of the CMLAN weights with a common input space were trained to handle discontinuity caused by two different target configurations. Each CMLAN set has two sub-CMLAN joint movement controllers resulting in one CMLAN input and two CMLAN trained outputs. Four differnt learnings such as batch SEC(G=0.05), on-line SEC(G=0.8), MEC(G=0.6), and REC(G=0.6), and REC(G=0.6) were applied for the equivalent training period.

The measured cup times on VAX 11/750 were 23.1 seconds and 24.8 seconds per learning epoch for batch and on-line type SEC learnings respectively. It took 15.18 seconds per epoch and 0.074 seconds per each random learning for MEC and REC respectively. The quantizing value K and the offset of the CMLAN input space were set to be 150 and 1. The interval of the sampled input nodes were 20 mm and 20 degrees resulting in 336 input pairs.

The rms and maximum errors were obtained from the joint and world spaces and are shown in fig. 6 and fig. 7 respectively. In the joint space, the rms and maximum errors were computed from the combined erros in joint one and joint two. In the world space, those are computed based on the Euclidian distance. The initial rms and the maximum

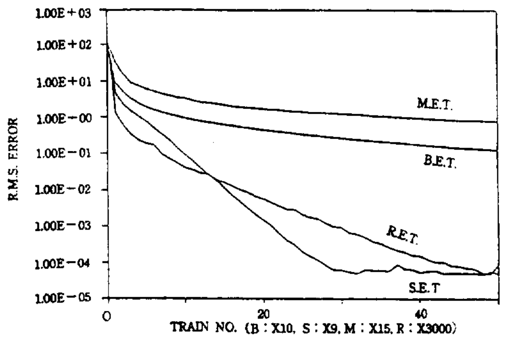


a) initial arm state and workspace

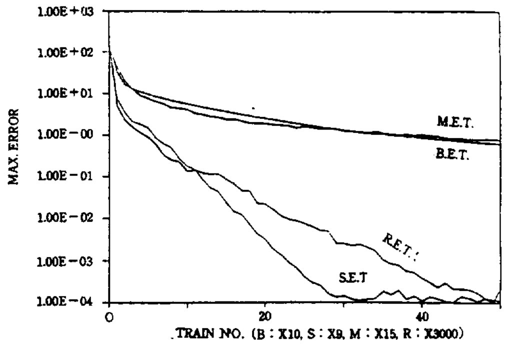


b) arm configuration

Fig. 4. Two different target configurations of two dof manipulator.



(a) rms error



(b) maximum error

Fig. 6. The performance in joint space using various learnings over sampled input nodes : (a) rms error, (b) maximum error (Sampled interval=20 deg and 20 mm, Batch SEC : G=0.05, On-line SEC : G=0.8, MEC : G=0.6, REC : G=0.4).

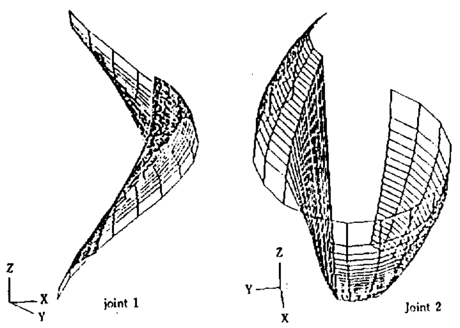
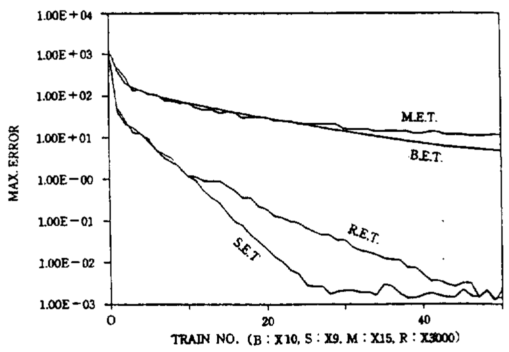
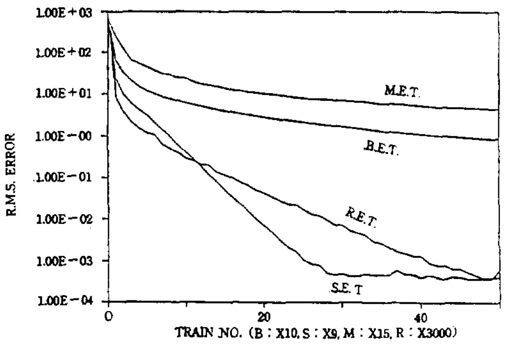


Fig. 5. Joint movements to be trained with $300<r<700$(mm) and $-150<\phi<150$ (deg).



(a) rms error



(b) maximum error

Fig. 7. The performance in world space using various learnings over sampled input nodes : (a) rms error, (b) maximum error (Sampled interval=20 deg and 20 mm, Batch SEC : G=0.05, On-line SEC : G=0.8, MEC : G=0.6, REC : G=0.6).

errors in the joint space were 85.07 deg and 141.5 deg, respectively. And in the world space, 782.7 mm and 1347 mm. The learned results from each learning scheme are shown in table 1.

The oscillating behavior of the rms error occurs in the batch SEC in the world space at the early stage although it is not shown clearly in Fig.7. The reason is mainly due to the unmatched relation of converging directions between the joint space and the world space.

The SEC and REC learnings exhibit the excellent trained performance while the MEC shows a poor result as expected. However, if the input pairs are increased, the required training cpu time will be the problem as mentioned earlier as a restriction of the SEC learnings.

Table 1. List of errors on each learning scheme (unit : deg and mm)

Error	Algorithm	Batch SEC	On-line SEC	MEC	REC
RMS	Joint	9.08E−2	3.54E−5	5.49E−1	6.72E−5
	World	8.24E−1	3.96E−4	4.45E−0	6.32E−4
MAX	Joint	6.27E−1	1.12E−4	7.93E−1	2.27E−4
	World	4.79E−0	1.48E−3	1.17E+1	2.40E−3

Robot inverse kinematic control problem has been also attempted to solve by applying or modifying the well-known multi-layer network under the generalized back-propagation(BP) delta-rule(14−17). Trained results from BP are relatively poor compared to the results presented in this paper though they require a smaller memory requirement. For instance, Guez(14) stated the BP trained results of two d.o.f manipulator was not adequate as a substitute for the closed form inverse kinematic solution because of the too much discrepancies. However, CMLAN based results under on-line SEC learning showed the maximum joint error of 1.12E−4 degree and the maximum workspace error of 1.48E−3 mm over the sampled nodes, which are good enough to be applied for a direct substitute. Besides, as the price of memory gets cheaper, the capacity of system memory is not a serious problem to a certain degree.

4. CONCLUSION

CMLAN is a kind of one-layer linear associator with a linear activation function having linearly independent binary pattern vector inputs. With proper number of sampled input node inputs, CMLAN can learn the desired nonlinear system behavior arbitrarily closely. The performance of the CMLAN based learning was quite good enough to implement the memory driven control system beacause of the enormously small size of the required system memory compared to the normal table look-up type storage.

As it is expected, CMLAN successfully generates the proper desired functional values to the nonlinear multi-variable function and to the joint commands of inverse kinematic motion without any prior knowledge in system modeling. Two d.o.f. inverse kinematic problem can be easily extended to six or more d.o.f. problem by increasing sub-CMLAN corresponding to each joint. Discontinuity can be handled by constructing two separate sets of CMLAN structures with suitable descision making net. Decision making can be done using an ordinary back-propagation net or CMLAN by utilizing a sigmoid type nonlinear activation function instead of unity activation function.

The presented results of the CMLAN analysis on learning will accellerate and extend its engineering application to the various fields especially for the system of the interest having a difficulty in its precise modeling, for the control system requiring the real time adaptation, and for the information processing like sensor fusion from the vision and other instruments readings. Application of CMLAN toward the visual image recognition and obstacle avoidance is undergone by the authors.

Since the effects of the control parameters on the CMLAN learning algorithm are critical, interrelations between these parameters and the shapes of the system function to be trained should be investigated with its learning performance. In the case that the unknown desired relations between the input and output are to be trained, it is desirable to set a certain guideline for the application of the CMLAN. Results of research performed related to the CMLAN design guide toward the efficient CMLAN application will be presented later.

REFERENCES

1. Albus, J.S. Dec 1972. Theoretical and experimental aspects of a cerebellar model. Ph.D. Thesis, Univ. of Maryland.
2. Albus, J.S. Sept. 1975. A new approach to manipulator control : The cerebellar model articulation controller (CMAC). Journal of Dynamic Systems, Measurement, and Control, Trans. of the ASME, Vol. 97, No. 3 : 220−227.
3. Albus, J.S. Sept. 1975. Data storage in the cerebella model articulation controller (CMAC). Journal of Dynamic Systems, Measurements, and Control, Trans. of the ASME, Vol. 97, No. 3 : 228−233.
4. Albus, J.S. 1979. Mechanisms of planning and problem solving in the brain. Mathematical Biosciences, 45 : 247−291.
5. Albus, J.S. June 1979. A model of the brain for robot control. BYTE Magazine : 54−95.
6. Camana, P.C. 1977. A study of physiologically motivated mathematical models for human postural control. Ph.D. Thesis, Dept. of Electrical Eng., Ohio State Univ.
7. Rajadhyaksha, J. 1985. A 2−D adaptive multilink CMAC manipulator simulation. MSME Thesis, Dept. of Mechanical Eng., Louisiana State Univ.
8. Manglevhedakar, S. 1986. An adaptive hierarchical model for computer vision. MSME Thesis, Dept. of Mechanical Eng., Louisiana State Univ.
9. Miller III, W.T. April 1987. Sensor-based control of robotic manipulators using a general learning algorithm. IEEE Journal of Robotics and Automation, Vol. RA−3, No. 2 : 62−75.
10. Miller III, W.T., F.H. Glantz, and L.G. Kraft, Summer 1987. Application of a general learning algorithm to the control of robotic manipulators. Int. Journal of Robotics Research Vol. 6, No. 2 : 84−98.
11. Rumelhart, D.E. and J.L. McClelland. 1987. Parallel distributed processing : Explorations in

the microstructure of cognition. Vol. 1&2, MIT Press, Cambridge.

12. Anderson, J.A. and E. Rosenfeld. 1988. Neuro-computing : Foundations of research. MIT Press, Cambridge, MA.
13. Pao, Y.H. 1988. Adaptive pattern Recognition and Neural Networks, Addison-Wesley Publishing Co. Inc., 1988, pp. 193－220.
14. Guez, A. and Z. Ahmad. June 1988. Solution to the Inverse Kinematic Problem in Robotics by Neural Networks. Proc. of the IEEE 2nd Int. Conf. on Neural Networks, Vol. II, pp. 617－624.
15. Josin, D. and et. al. Sept. 1988. Robt Control Using Neural networks. Proc. of the Int. Neural Network Society 1st Annual Meeting.
16. Sobajic, D.J. and et al. June 1988. Intelligent Control of the Intelledex 605T Robot Manipulator. Proc. of the IEEE 2nd Int. Conf. on Neural Networks, Vol. II, pp. 633－640.
17. Elsley, R.K. Sept. 1988. A Learning Architecture for Control Based on Back-Propagation Neural networks. Proc. of the Int. Neural Network Society 1st Annual Meeting.

祝

學 位 取 得

姓　　名	曹　永　吉
生 年 月 日	1948年 11月 12日
勤 務 處	農業機械化硏究所
取 得 學 位 名	農學博士
學 位 授 與 大學	서울大学校 大學院
學位取得年月日	1990年 8月 30日
學 位 論 文	自脫型 콤바인 還元裝置의 還元物 流動 現象과 還元 性能 改善에 關한 硏究