MINDSTORMS을 이용한 프로그래밍 학습이 창의력에 미치는 효과

유인환†·김태완††

요 약

기존 프로그래밍 교육은 내용과 방법, 대상에 대한 고려가 부족하였다. 특히, 초등학생에 적합한 교육은 거의 이루어지지 않아 교육현장에서 중요성이 점점 감소하게 되었다. 이러한 문제를 해결하고자 본 연구에서는 초등학생들에게 적합한 프로그래밍 교육 도구로서 MINDSTORMS의 활용을 제안하고, 학습자의 창의력에 미치는 효과를 검증하였다. 학교에서 적용 결과 학생들의 창의력 신장에 많은 도움을 주는 것으로 나타났다. 프로그래밍 교육의 목적은 단지 프로그래밍 언어의 이해가 아니라 문제해결력, 논리적 사고력, 창의력 신장에 두어야 한다. 또한, 학생들이 스스로 그들의 학습활동을 통제하고 능동적으로 문제를 해결할 수 있는 환경을 조성해 주어야 하며, 이러한 프로그래밍 교육에 MINDSTORMS은 매우 유용하고 적절한 도구가 될 수 있다.

키워드 : 마인드스톰, 프로그래밍 교육, 로봇 프로그래밍

The Effects of MINDSTORMS Programming Instruction on the Creativity

In-Hwan Yoo†·Tae-Wan Kim††

ABSTRACT

Traditional programming education lacked consideration of contents, methods and learners. In particular, the importance of programming education has been decreased in school because almost no suitable teaching has been executed for elementary school students. To solve these problems, this study proposed the use of MINDSTORMS that is a programming education tool suitable for elementary school students, and verified its effects on learners' creativity. The results of this study show that the tool is highly effective in improving students' creativity. The objectives of programming education are not only learning programming languages but also improving problem-solving ability, logical thinking and creativity. We must offer environment, in which students can control their own learning activity and solve problems by themselves. In addition, MINDSTORMS can be a very useful and suitable tool for programming education.

Keywords : MINDSTORMS, programming education, robot programming

1. 서 론

프로그래밍의 교육적 가치는 전문가들 사이에서 계속 논의되어 왔다. 그 중에는 컴퓨터 언어를 사용하여 프로그래밍을 한다는 것은 너무 어렵고 복잡하여 문제 해결력 신장에 효과가 없다고 주

†종신회원: 대구교육대학교 컴퓨터교육과 교수(교신저자)
††정 회 원: 대구효성초등학교 교사
논문접수: 2005년 10월 27일, 심사완료: 2005년 12월 9일

장하는 연구도 있으나, 대부분은 컴퓨터 프로그래밍 학습이 문제 해결능력에 긍정적인 영향을 미친다고 보고하고 있다[7].

프로그래밍은 코딩(coding) 과정에서 문제분석력과 이해력을 기르고, 코드를 분석하는 과정에서 논리적 사고력을, 오류 검증 및 수정작업에서 반성적 사고력과 같은 고등인지기술을 향상시킬 수 있으며, 이러한 과정에서 컴퓨터를 더 깊이 이해할 수 있고, 이를 바탕으로 컴퓨터를 보다 잘 활용할 수 있는 기초를 닦을 수 있다.

그러나 현재의 우리나라 교육환경에서는 프로그래밍 교육이 매우 어려운 실정이다. 국민공통기본교육과정에서 프로그래밍 교육이 제외되어 있어 정책적으로 소외되어 있기 때문이다. 이는 프로그래밍 언어 자체가 어려워 배우거나 가르치기 힘들며, 교육용 프로그래밍 언어의 선정과 구매의 어려움과 이를 지도할 교사의 역량의 부족 등 현실적인 난제가 많기 때문이기도 하지만, 가장 본질적인 문제는 프로그래밍 교육의 효과에 대한 의문과 K12 학생들에게 알맞은 프로그래밍 교육 내용이나 방법의 개발에 대한 연구가 부족했다는 점에 기인한 것으로 추정된다.

이러한 문제의식을 가지고 본 연구에서는 프로그래밍 교육에서 로봇의 활용을 제안하고, 그 의미와 가능성에 대해 논의하고자 한다.

외국의 경우 로봇을 교육적으로 활용하고자 하는 탐색이 다양하게 시도되고 있다. Fagin, Merkle, Eggers(2001) 등은 로보틱스를 이용하여 컴퓨터 과학을 가르치는 연구를 수행했는데, 로봇의 사용은 프로그래밍 경험이 없는 학생들에게 직관적이고 손으로 느끼는 학습 경험을 제공하였으며, 설계-코딩-실행-재설계의 피드백 과정이 매우 빠르게 진행되었을 뿐만 아니라, 학생들이 이 과정을 즐겁게 받아들여 로봇을 사용하는 과정과 그렇지 않은 과정에서는 많은 차이가 나타나는 것으로 보고하고 있다[14].

로봇을 이용한 프로그래밍 관련 연구에서 공통적으로 제시되는 결과는 로봇 제작에 대부분의 학생들이 흥미를 느끼며, 특히 만들어진 로봇의 시연과 오류 수정에 큰 기쁨을 느끼고, 적극적인 참여가 이루어지고 있다고 보고하고 있다 점이다[13][14][15][21].

외국에서 로봇의 교육적 활용 연구 사례에서 많이 사용되는 도구는 MINDSTORMS이다. 이는 초등학생에게 적합하게 설계된 교육용 완구로서 프로그래밍이 가능한 컴퓨터와 브릭(bricks)으로 구성되어 있으며, 로보틱스에 대한 전문적인 지식을 갖추지 않아도 센서와 부품을 사용하여 쉽게 로봇을 제작하고 프로그래밍할 수 있기 때문에 교육적 활용 가능성이 매우 높다.

MINDSTORMS의 활용에 대한 연구는 주로 공과대학의 프로그래밍이나 로보틱스의 입문과정에서 다루어지고 있었으며, K12에서의 활용은 과학, 수학 등의 교과에서 이용하는 것이 대부분이었다. 그러나 초등학생 대상의 프로그래밍 교육에서 MINDSTORMS을 활용하는 연구는 찾기 힘들었다. 더욱이 우리나라의 경우에는 MINDSTORMS의 교육적 활용에 대한 연구 자체가 거의 없는 실정이다.

이에 본 연구에서는 프로그래밍 교육의 내용과 방법으로서 MINDSTORMS의 활용을 제안하고자 한다. MINDSTORMS을 이용하는 프로그래밍 수업모형과 내용을 개발하고, 학교현장에서 적용하여 학습자의 창의력에 미치는 효과를 검증하고자 한다.

2. MINDSTORMS의 이해

2.1 MINDSTORMS의 개념

MINDSTORMS은 덴마크 LEGO社(The LEGO Group)와 미국 M.I.T대학이 공동으로 개발한 교육완구의 일종으로 마이크로월드(microworlds)의 개념을 로보틱스(robotics)와 접목시킨 것이다. 기존 브릭(brick)형 완구와의 가장 큰 차이점은 컨트롤러 브릭(controller brick), 즉 마이크로 컴퓨터(micro computer)가 추가되어 프로그래밍이 가능해졌다는 점이다.

이 프로그래밍이 가능한 장난감은 PC를 이용하여 프로그래밍한 후 적외전 전송장치를 통해 RCX(Robotics Command System)라는 마이크로 컴퓨터로 전송받음으로써 생명력을 얻는다[17].

MINDSTORMS은 로봇을 만들거나 자동화, 상

호 작용이 가능한 시스템을 만들기 위한 프로그램 가능한 브릭(bricks)과 모터, 센서, 기어, 축 등의 부품들을 포함하고 있다. MINDSTORMS은 공학적인 장남감이기도 하지만, 컴퓨터로 제어되는 교육적 도구로서, 전기공학적인 부품으로 만들어진 임베디드 시스템(embedded system)의 좋은 예이기도 하다. MINDSTORMS을 이용하여 엘리베이터 컨트롤러부터 산업용 로봇까지 실생활의 거의 모든 종류의 임베디드 시스템을 모델링할 수 있다[20].

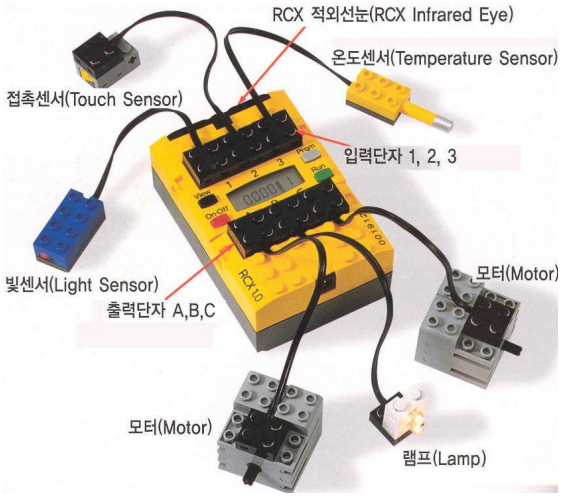
2.2 MINDSTORMS의 구성요소

MINDSTORMS은 일반적인 LEGO 브릭과 더불어 프로그래밍을 위한 몇 가지 요소가 추가되어 이루어져 있는데 세부 내용은 다음과 같다.

- ·RCX: RCX는 마이크로 컴퓨터로 MINDSTORMS의 두뇌역할을 하는 브릭이다. 8비트 CPU에 LegOS라는 운영체제를 탑재하고, ROM과 RAM 그리고 입출력 포트를 각각 3개씩 가지고 있다. RCX는 다섯 개의 프로그램을 저장할 수 있고, PC에서 작성된 코드를 적외선 전송장치를 통해 다운로드 받아 RUN 버튼을 눌러 선택한 프로그램을 실행하게 된다. <그림 1>은 RCX의 입출력 단자에 센서와 모터, 램프를 장착한 모습이다.
- ·LEGO 부품: 로봇의 몸체를 만들 수 있는 기본 재료로 브릭, 기어, 축, 전선 등이 제공되며, 보통 RCX가 하나의 큰 브릭으로 몸체를 형성한다. 입력장치로는 접촉센서(touch sensor), 빛센서(light sensor), 회전센서(rotation sensor), 온도센서(temperature sensor) 등이 있으면 출력장치는 모터(motors), 램프(lamp) 등이 제공된다.
- ·적외선 전송장치(IR transmitter): PC와 연결하여 사용하는데, PC에서 프로그램된 코드를 전외선 방식으로 RCX로 전송해준다.
- ·프로그래밍 도구: RCX는 일반적인 컴퓨터 언어나 전용 프로그래밍 언어를 이용하여 프로그래밍 할 수 있다.

2.3 마이크로월드로서 MINDSTORMS

마이크로월드는 컴퓨터 프로그램의 상황 하에서 실세계의 영역과 관련된 대상에 대해 정의를 내릴 수 있는 세계를 구성하고자 하는 목적으로 고안되



<그림 1> 입출력 장치를 장착한 RCX

었다. 실세계에 해당하는 부분은 대부분 매우 복잡했기 때문에 초기의 마이크로월드는 실재를 단순화하고, 행동(action)에 관한 이론을 테스트하는 실험의 역할을 했다. Papert(1980)는 교육적 문맥에서 사용될 몇 가지 마이크로월드를 발전시켰으며 Piaget의 인지적 발달이론으로 이를 확고히 했다. 마이크로월드는 학습을 돕기 위해 설계되었으며 '가르침 없이 배우기'라는 교육적 철학을 내포하고 있는데, 로고(LOGO) 프로그래밍 환경이 그 대표적 예이다[1].

또한, 마이크로월드에서는 '문제'에 초점을 두고 있는 것이 아니라 '적절한 현상' - 학습을 유의미하게 일어날 수 있도록 보조하는 흥미로운 현상들 - 을 관찰하거나 발견해 내는데 초점이 있다. 즉, 학습 동기가 유발되고 학습에 필요한 자료가 풍부하여 학습이 유의미하게 일어나도록 학습 환경을 인위적으로 조성하여 제공하는 것이 마이크로월드이다[6].

결국 마이크로월드란 실세계를 단순화시켜 가상공간에 표현한 모형으로, 학생들이 자신의 수준에 맞게 대상을 조작하고 탐구함으로써 의미 있는 개념과 원리들을 스스로 이끌어 낼 수 있는 학습 환경을 의미한다. 본 연구에서 프로그래밍 도구로 활용하는 MINDSTORMS은 학습자들이 로봇을 통해 실세계를 모델링하고 프로그래밍을 통해 이를 조작하는 탐구활동을 통해 개념과 원리들을 학습하게 되므로 로보틱스를 이용해 마이크로월드를 구현한 하나의 좋은 예라고 할 수 있다.

2.4 MINDSTORMS 이론적 배경

Papert(1980)는 구성주의적 입장에서 지적인 메타포(metaphor)로 컴퓨터 프로그래밍 도구인 MINDSTORMS를 개발하였다. 그는 학습이란 학습자 자신의 지식 체계에 외부의 자극을 나름대로의 체계로 재구성해 가는 과정이라고 보는 구성주의적 관점에서 교육을 이해하고, 그와 같은 관점에서 아동의 사고 체계를 구성하는데 MINDSTORMS이 학습 환경을 제공할 수 있다고 주장하였다. Papert는 환경과의 상호 작용에 의해 사고 체계가 구조화되어 간다는 점에서 Piaget의 영향을 많이 받았으나, 인지 발달 단계의 생물학적 제한을 뛰어 넘을 수 있고, 이것은 적절한 교육 환경을 제공함으로써 가능하다고 보는 관점에서는 다르다. 그는 환경과 사고 체계의 의미 있는 상호 작용을 촉진시켜 생물학적 인지발달 단계의 제한을 뛰어 넘을 수 있다고 보고 그것은 MINDSTORMS이라는 개방적 학습 환경으로 가능하다고 하였다[18].

Piaget에 의하면 지각할 수 있는 대상으로부터 그 공통 성질을 이끌어 내는 것을 경험적 추상화라고 부르고, 행동과 조작의 일반적인 조정으로부터의 추상화를 반영적 추상화라고 부르고 있다. 반영적 추상화는 모든 논리, 수학적 지식 획득의 심적 메카니즘이며, 모든 논리, 수학적 개념은 주체 자신에 의해서 구성된 것이라고 주장하였다. 아동은 자신의 활동과 그 결과에 대한 경험을 통해 개념을 추상화한다. 이 과정에서 매우 중요한 점은 대상에 대한 구체적 경험 이상으로 행동과 그 결과에 대한 반성을 통해 내적 재구성이 이루어진다는 점이다.

Papert는 Piaget의 인식론에 근거하여 지식의 학습에서 아동의 활동과 주위 환경과의 능동적인 상호 작용을 중요하게 생각하며 '활동을 통한 학습'을 강조한다. 아동들이 자기 수준에서 자신의 사고 구조에 적절한 방식으로 스스로 새로운 발전을 하는 자기 생성 학습을 강조한다. MINDSTORMS 프로그래밍 교육의 교육적 핵심 요소는 자기의 사고를 의식화시킨다는 점과 아동들에게 인위적이면서도 자연스러운 학습 환경을 제공한다는 점이다. Piaget는 아동의 지적 발달은 주로 반영적 추상화에 의해서 이루어진다고 주장한다. 이 반영적 추상화는 자기 자신의 행동을 높은 차원으로 조직하는 것으로 프로그래밍 도구로서의 MINDSTORMS의 핵심적인 이론적 배경이다. 즉, 로봇의 제작을 통해 개념과 원리를 추상화하고, 이러한 자기 자신의 행동을 반성하며, 다시 로봇을 동작시키기 위하여 프로그래밍하고 그 결과를 다시 수정, 보완하며, 더 발전된 것을 생각해 내는 것이다. 학습자 자신이 직접 LEGO 브릭을 조작하고 프로그래밍하며, 그 행동을 반성하여 추상화한다는 아이디어는 '자신의 행동에 대한 반성으로부터의 추상', 즉 반영적 추상화와 일맥상통한다.

3. MINDSTORMS 프로그래밍 교육

3.1 수업모형

프로그래밍 수업방법의 유형에 대한 검토를 통해 MINDSTORMS 수업에 적당한 모형을 탐색해 보기로 한다. 프로그래밍 학습의 목적이 단순히 언어의 문법 습득이 아니라 학습자의 창의력과 문제해결력을 증진시키는데 있다고 본다면 MINDSTORMS의 경험이 다른 상황에도 전이되기 쉽도록 가르쳐야 한다. 따라서 고안하고자 하는 수업모형은 이러한 점이 잘 고려되어야 한다.

프로그래밍 수업방법의 유형으로 Littlefield 등(1988)은 비구조적(unstructured)수업, 구조적(structured)수업, 중재적(mediated) 수업방법으로 구분하였고, Delclos 등(1984) 은 발견학습(discovery learning), 구조적 교수(structured tutorials), 중재적 형식의 수업(teaching in the a mediational style)으로 구분하였다. 또한 Roach(1988)는 안내된 발견식(guided discovery approach)과 직접적(direct)수업방법으로 나누었다[4]. 조미옥(1991)은 안내적 교수법은 명백한 교수설계(explicitly modeled instructional design)와 중재적 학습을 결합한 개념이라고 하며, 학생들이 의식적으로 지식, 정보 또는 학습전략을 습득해 나가도록 도울 뿐 아니라, 학생들에게 습득된 지식 및 전략을 다른 새로운 문제 상황에 전이할 수 있도록 해준다고 말하고 있다[9].

Papert(1980)는 Piaget의 영향을 받아

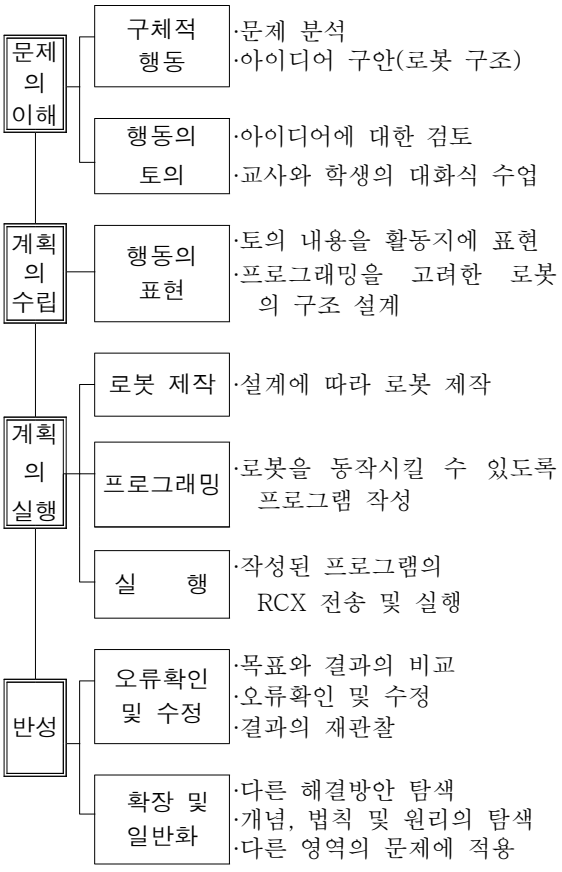
‘MINDSTORMS : Children and computers and powerful ideas’라는 그의 저서에서 ‘교육과정 없는 학습(learning without curriculum)’, ‘교수 없는 학습(learning without teaching)’을 주장했다. 즉 Papert는 구조화가 적게 된 수업방법일수록 학습자에게 더 많은 것을 제공한다고 했다. 아이들은 자신들의 학습을 스스로 지도하고 통제할 수 있다고 주장한다. 그러나 실제로 아이들은 스스로 학습할 수 있다는 것을 체험하고 이를 확장시키는 데 익숙해 있지 않다[19].

그러므로 학생들에게 프로그래밍 활동에 적극적으로 참여할 수 있도록 교사의 최소한의 개입이 필요하게 된다. 따라서 본 연구에서는 안내의 정도에 따라 수업방법을 구분한 안내된 발견식 수업(guided discovery teaching method)모형을 적용하는 것이 적절하다고 판단하였다.

안내된 발견식 수업은 학생들이 프로그래밍 활동을 능동적으로 할 수 있는 환경을 조성하여 주되, 교사가 문제해결 과정에 따라 학습 진행 과정을 모니터링 하여 학생들의 창의력을 신장시키는데 목표를 두고 있다. 이러한 학습 환경은 학생들의 문제해결과정을 활성화시킴으로써 학생들을 동기화시키는데 결정적이다. 안내된 발견적 수업은 교사가 일방적으로 지식이나 정보를 전달하는 교사중심의 지시적 수업이나 학생들의 무작위적인 활동에만 학습의 형태를 모두 배제한다. 즉 이 수업방법에서는 교사와 학생들이 모두 중요한 의미를 가진다. 안내된 발견식 수업에서 교사는 새로운 학습전략 정보를 전달하는 데 있어 소크라테스의 대화법(socratic dialogue)을 사용한다[9].

이는 학생들 스스로가 의식적으로 주어진 문제상황에 관심을 갖도록 유도하기 위한 것이다. 구체적인 답은 주어지지 않으나 학생들이 방향을 찾지 못하고 방황할 때 유도적 질문으로 학생들이 가야할 방향을 제시한다. 소크라테스의 대화법은 학생들의 사고를 기억 위주에 머물게 하기보다는 능동적, 역동적으로 활성화시키는데 도움을 준다. 이와 같은 안내된 발견식 수업은 학생들이 의식적으로 지식, 정보 또는 학습전략을 습득해 나가도록 돕고, 습득된 지식 및 전략을 다른 새로운 문제상황에 전이할 수 있도록 해준다.

본 연구자는 백영균(1994)의 모형[4]을 보완하여 수업모형을 개발하였다. 구체적인 지도단계와 내용을 도식화 하면 <그림 2>와 같다. 이 수업모형은 MINDSTORMS을 이용하는 프로그래밍 교육에서 활용할 수 있는 일반적인 모형으로 개발되었으나, 본 연구에서 모든 차시에서 적용하기는 어려움이 있었다. 1차시 단위로 수업을 진행할 수밖에 없었던 실험의 시간적 여건 때문이기도 하며, 초기 수업에서는 주로 MINDSTORMS, ROBOLAB의 내용 소개와 제시된 예제의 연습에 초점을 맞추었기 때문이다(<표 1> 차시별 학습내용 참조). 본 연구의 실험에서는 본격적인 프로그래밍 학습 단계인 16차시의 INVENTOR 프로그래밍 예제로 실습하기부터 본 모형이 본격 적용되었다.



<그림 2> 수정된 안내된 발견식 수업모형

3.2 학습내용의 선정

MINDSTORMS의 RCX는 C, Java, Visual Basic 등 일반적인 프로그래밍 언어를 이용하여 프로그래밍 할 수 있다. 이런 언어들은 매우 강력한 기능을 제공하지만 프로그래밍에 대한 많은 훈

련과 지식을 필요로 한다. 이에 비해 ROBOLAB은 누구나 쉽게 사용할 수 있는 간단한 환경의 프로그래밍 도구이다. 1997년 NASA가 화성에 소저너(Sojourner)를 착륙시켰을 때 National Instrument 사의 LabVIEW라는 소프트웨어를 사용하여 이를 제어하였었다. ROBOLAB은 이 LabVIEW에 기초를 두고 RCX를 프로그래밍하기 위해 만들어진 소프트웨어이다. 본 연구의 대상이 초등학생인 점을 감안하여 어린이들도 쉽게 프로그래밍 할 수 있는 ROBOLAB을 RCX 프로그래밍 도구로써 적절하다고 판단하여 이를 사용하였다.

위에서 개발한 수업모형에 따라 실험을 실시하기 위하여 <표 1>과 같이 MINDSTORMS과 ROBOLAB을 이용한 프로그래밍 학습내용을 선정하였다.

<표 1> 차시별 학습내용

차시	학 습 내 용
1	MINDSTORMS의 개념과 구성요소, 사용법을 통해 프로그래밍을 통해 로봇 동작시키기
2	ROBOLAB을 이용한 프로그래밍으로 프로그래밍에 대한 동기부여 및 프로그래밍의 원리 발견하기
3	ROBOLAB PILOT 소개 및 프로그래밍 기초단계: LEVEL 1 연습 - 프로그램을 통해 RCX의 출력단자 하나의 전원 공급량 조절하기 - 시계모양의 아이콘을 이용하여 지정된 시간 동안 모터 움직이기
4	PILOT 프로그래밍: LEVEL 2 연습 - 프로그램을 통해 RCX 출력단자 2개의 전원 공급량 조절하기
5	PILOT 프로그래밍: LEVEL 2 연습 - 특정한 시간 동안 또는 입력단자에 연결된 센서를 통한 전원 공급 조정하기
6	PILOT 프로그래밍: LEVEL 3 연습 - 프로그램을 통해 RCX의 출력단자 3개 모두 사용하여 전원 공급 조절하기
7	PILOT 프로그래밍: LEVEL 4 연습 - 프로그램의 길이를 늘여 다양한 과정의 프로그래밍 작성하기
8	PILOT 프로그래밍: LEVEL 4 연습 - 센서를 이용하여 모터의 반응을 다양하게 설정하여 프로그래밍 작성하기
9	통합 개발 환경 익히기를 통해 프로그래밍 기초 기술 익히기
10	ROBOLAB INVENTOR 프로그래밍을 통해 제약이 줄어든 프로그래밍 작성해 보기
11	ROBOLAB INVENTOR 프로그래밍 고급단계: LEVEL 1 연습 - 명령 아이콘의 나열을 통한 프로그래밍 작성하기 - 나열식 프로그래밍 작성하기
12	INVENTOR 프로그래밍: LEVEL 2 연습 - 연결단자 설정이나 전원 공급량 설정과 같은 MODIFIER를 가지도록 프로그래밍 작성하기
13	INVENTOR 프로그래밍: LEVEL 3 연습 1 - 높은 수준의 논리 회로를 만들 수 있도록 프로그래밍 작성하기 - 조건검색 실행, 다중작업, 반복실행의 프로그래밍 작성하기
14	INVENTOR 프로그래밍: LEVEL 3 연습 2 - 음악 메뉴를 이용하여 모터나 램프의 상태에 따라서 다른 음을 내도록 프로그래밍 작성하기
15	INVENTOR 프로그래밍: LEVEL 4 연습 - 전 단계의 명령 아이콘과 MODIFIER를 모두 사용하여 프로그래밍 작성하기 - RCX와 RCX간의 데이터 전송이 가능한 프로그래밍 작성하기
16	INVENTOR 프로그래밍: LEVEL 4 예제로 실습하기 - FUNCTION창의 각종 메뉴 등 각종 아이콘을 사용하여 폭넓은 프로그래밍 작성하기
17	INVENTOR 프로그래밍 예제로 실습하기 : BEEPS 실습예제를 통해 익히기
18	INVENTOR 프로그래밍 예제로 실습하기 : CONTAINERS 실습예제를 통해 익히기
19	INVENTOR 프로그래밍 예제로 실습하기 : FORKS 실습예제를 통해 익히기
20	INVENTOR 프로그래밍 예제로 실습하기 : MOTORS & LIGHTS 예제를 통해 익히기
21	INVENTOR 프로그래밍 예제로 실습하기 : TASK SPLITS 실습예제를 통해 익히기
22	INVENTOR 프로그래밍 예제로 실습하기 : TIMERS 실습예제를 통해 익히기
23	INVENTOR 이용한 응용 프로그래밍 1 : 나만의 프로그래밍 작성하기
24	INVENTOR 이용한 응용 프로그래밍 2 : 상호간 문제를 제시하여 프로그래밍 작성하기

1, 2, 3차시에서는 MINDSTORMS을 이용하여 간단한 로봇을 직접 제작하며 그 절차와 방법에 익숙해지도록 하였다. 이 단계에서 주된 수업 내용은 로봇 제작과 프로그래밍 절차에 익숙해지도록 하는 것으로써 구체적 내용은 다음과 같다. 첫째, 설계 단계에서 문제를 분석하고 계획을 수립하여 만들고자하는 로봇을 설계하며, 둘째, 조립 단계에서 LEGO 부품을 이용하여 로봇을 제작하고, 셋째, 프로그래밍 단계에서 PC에서 조립된 로봇이 동작할 수 있도록 프로그램을 작성하며, 넷째, 전송 단계에서 PC에 연결된 적외선 전송장치

를 통해 RCX에 코드를 전송하여, 다섯째, 실행 단계에서 RCX의 RUN 버튼으로 프로그램을 실행시켜 로봇이 동작하도록 한다.

1차시는 로봇 프로그래밍에 대한 개요 소개이며, 2차시부터는 ROBOLAB을 이용한 프로그래밍을 시작하였다. ROBOLAB을 이용한 RCX 프로그래밍은 PILOT과 INVENTOR라는 두 가지 단계를 제공한다. PILOT 프로그래밍은 RCX 프로그래밍의 기초 단계로 주어진 형식에 맞추어 명령 아이콘을 선택하는 방식이다. PILOT 프로그래밍은 다시 네 개의 LEVEL로 정규화된 형태로 제공된다. LEVEL1에서 LEVEL 4로 갈수록 복잡해진다. 네 단계는 모두 체계적으로 연결되어 있으므로, 전 단계를 이해하는 것은 다음 단계를 시작하는데 많은 도움이 된다.

이와 같이 ROBOLAB은 하나의 프로그래밍 도구이자 그 자체가 일종의 코스웨어처럼 프로그래밍을 단계적으로 학습할 수 있도록 LEVEL 단위로 예제가 제공되므로, 본 실험은 이를 기본으로 삼고 내용을 추가, 수정, 보완하여 <표 1>과 같이 24차시 학습을 진행시켰다.

본격적인 프로그래밍 교육은 10차시 INVENTOR 프로그래밍부터 시작하였다. INVENTOR 프로그래밍은 FUNCTION 창에서 아이콘을 선택해서 DIAGRAM 창으로 옮긴 다음 선으로 연결하여 프로그램을 만든다. INVENTOR 프로그래밍도 4개의 LEVEL로 이루어져 있으며, LEVEL1에서 LEVEL4로 갈수록 복잡해지는 대신 프로그래밍의 제약은 줄어든다. 그만큼 학습자는 창의적으로 설계하고, 조작해서 프로그래밍을 할 수 있게 된다.

15차시까지는 INVENTOR의 연습 단계이며 16차시부터 본격적으로 프로그래밍 학습이 시작된다. INVENTOR LEVEL 4는 프로그래밍 과정 중에서 가장 높은 단계의 프로그래밍이다. 전 단계의 명령 아이콘과 MODIFIER를 모두 사용하여 RCX와 RCX간의 데이터 전송도 가능하다. 또한 컨테이너를 사용하여 수치를 할당할 수 있고 조작할 수도 있다. 이 단계에서는 프로그램을 짜는데 제한이 없기 때문에 여러 예제를 통해 학습을 진행시켰다. 끝으로 23, 24 차시에서는 학생들이 자신이 스스로 자유롭게 설계한 로봇을 제작하고 이에 따라 프로그래밍할 수 있도록 하였다.

4. 적용 및 분석

4.1 창의력과 검사도구

창의력 연구의 개척자라고 할 수 있는 Guilford(1967)는 창의력의 개념을 확산적 산출(divergent production)을 사용하여 지식의 변환(transformation)을 일으키는 과정이라고 설명하고, 구성 요소를 문제에 대한 민감성(sensitivity to problems), 사고의 유창성(fluency of thinking), 사고의 융통성(flexibility of thinking), 사고의 독창성(originality of thinking), 사고의 정교성(elaboration of thinking), 재정의 및 재구성(a factor involving reorganization or redefinition of organized whole)을 들었다[3][16].

Torrance(1974)는 창의성을 어떤 문제, 결핍, 지식의 결여 및 부조화 등에 대하여 민감해짐으로 해서 그 상황에서 문제점을 찾아내고 문제 해결 방법을 탐색하여 결함에 대해 추측해 보고 가설을 세워 보며, 이러한 가설들을 검증 및 재검증을 거듭한 후 그 결과를 타인에게 전달하는 전과정으로 보았다. 그는 창의적 사고의 주요 관점을 유창성, 융통성, 독창성, 정교성 등 4가지로 제시했다[2][11][22].

Williams(1972)는 창의적 사고의 관련 구인으로 유창성, 융통성, 정교성, 독창성을 들었고[4], Lewenfeld(1961)는 민감성, 유창성, 독창성, 재구성 능력, 분석, 결합, 조직력을 들고 있다. 창의력의 구성 요인과 관련하여 이상의 논의를 정리하면 <표 2>과 같다[12].

<표 2> 창의력의 구성 요인

구성요인 / 학자	유창성	융통성	독창성	정교성	기 타	
					재정의	조직력
Guilford	○	○	○	○	○	
Williams	○	○	○	○		
Torrance	○	○	○	○		
Lewenfeld	○		○		○	○

<표 2>에서와 같이 창의력의 구성 요인에 대한 여러 가지 견해가 있으나 유창성, 융통성, 독창성, 정교성 등을 공통요인으로 들 수 있다. 본 연구에서 이 네 가지를 창의력의 구성 요인으로

보고, 창의력은 “어떤 문제나 상황을 민감하게 지각하고, 이미 알고 있는 사실, 개념, 원리 등을 그 문제에 적용하여 기존의 관계 양식이나 해결 방법으로부터 탈피하여 다양하고 독특하게 문제를 해결해 가는 능력”이라고 정의한다.

창의력 검증은 Torrance(1974)의 “Test of Creative Thinking A, B"의 창의력 검사지[12][22]를 번안한 것을 사용하였다. 이 검사지는 A, B형 각각 도형형 3개 활동, 언어형 7개 활동으로 구성되어 있다. 그러나 본 연구에서는 검사 문항 모두를 사용하지 않고, 초등학교 6학년에 적합하도록 A, B형 각각 도형형 2개 활동, 언어형 4개 활동을 채택하여, A형을 사전 검사에, B형을 사후 검사에 사용하였다. 검사 실시 시간은 도형형 각 10분, 언어형은 각 5분씩의 제한 시간을 두었다.

이 검사지는 정원식의 표준화 간편 창의성 검사와 40인의 공인타당도를 가지며, 채점의 신뢰도는 .86 - .96, 동형검사 신뢰도는 .65이다[10][12]. 이 검사는 각 하위 검사마다 유창성, 융통성, 독창성, 정교성에 대해 채점한다. 이를 모두 합한 것이 창의력 총점이 된다.

4.2 실험 집단의 구성

본 연구는 OO광역시 H 초등학교 6학년 아동 중 2개 학급 60명을 임의적으로 A, B 집단으로 삼고 연구를 적용하였다. A, B 집단은 사전 실태조사 결과 비슷한 수준의 집단으로 분석되었다. 이들 중 A집단은 <표 1>과 같은 ROBOLAB을 이용한 MINDSTORMS 프로그래밍 교육을 아침 자습 시간(담임 재량 시간)에 24차시 동안 학습시키고, 다른 B집단은 특별한 처치를 가하지 않고 담임 재량에 따라 일반적인 교육활동을 실시하였다. 실험처치 후의 효과를 검증하기 위해 사전검사와 사후검사를 실시하였으며, 검사결과는 T-검증을 이용해 분석하였다.

4.3 실험 방법

본 연구의 실험 설계는 다음과 같다.

(G1)	O1	XG	O2
(G2)	O3	XS	O4

* G1 : MINDSTORMS 프로그래밍 수업 집단
* G2 : 일반수업 집단
* O1, O3 : 사전 창의성검사 (TTCT A형)
* O2, O4 : 사후 창의성검사 (TTCT B형)
* XG, XS : 실험처지

먼저 A, B 집단을 대상으로 사전 창의력 검사(TTCT A형)를 실시하고, 그 중 A 집단을 대상으로는 MINDSTORMS과 ROBOLAB을 이용한 프로그래밍 수업을 매일 40분씩 총 24차시 수업을 진행하고, 나머지 집단은 일반수업만 참여한 후 모두에게 사후 창의력 검사(TTCT B형)을 실시하였다.

4.4 실험 결과 및 분석

4.4.1 사전검사

본 연구의 가설들을 검증하기에 앞서, 실험집단과 통제집단에 따른 사전 창의력 검사 점수의 차이를 확인하기 위해 사전 창의력 검사(TTCT A형)를 실시하고, 그 결과를 T-검증을 통해 사전 창의력 검사에 대한 총점을 비교해 보았다.

<표 3> 사전 창의력 검사에 대한 총점 비교

구 분	학생수	평균	표준편차	t	df	p
실험집단	30	227.50	39.72	.490	58	.626*
통제집단	30	232.46	38.78			

*p < .05

<표 3>에 의하면, 사전 검사에서 평균은 실험집단이 227.50, 통제집단은 232.46로 나타났다. 또한 t 값이 0.490이고, 유의도가 0.626로 $p<0.05$ 수준에서 두 집단은 통계적으로 유의한 차이가 크게 없는 것으로 나타났다. 따라서 두 집단은 동질집단이라고 말할 수 있다.

4.4.2 사후검사

MINDSTORMS 프로그래밍 학습이 학습자의 창의력 신장에 미치는 영향을 조사하기 위하여 실험집단과 통제집단간의 창의력 총점에 대해서 독립표본 t-검증을 이용하여 분석해 본 결과 <표 4>와 같이 나타났다.

가설 1 : MINDSTORMS 프로그래밍을 학습한 집단이 학습하지 않은 집단보다 높은 창의력 신장을 보일 것이다.

<표 4> 사후 창의력 검사에 대한 총점 비교

구 분	학생수	평균	표준편차	t	df	p
실험집단	30	250.40	25.62	2.357	58	.021
통제집단	30	235.20	24.29			

*p < .05

두 집단 간의 사후 검사 자료를 t-검증으로 분석한 결과, 평균이 실험집단은 250.4, 통제집단은 235.2로 실험집단이 15.2 더 높게 나타났다. 또한 t 값이 2.357이고, 유의도가 0.021로 유의수준 0.05보다 작으므로 두 집단의 사후검사는 통계학적으로 유의한 차이가 있는 것으로 나타났다. 따라서 가설 1, 즉 MINDSTORMS 프로그래밍 학습은 창의력 신장에 영향을 주는 것으로 밝혀졌다.

창의력을 구성하는 유창성, 융통성, 독창성, 정교성 등의 하위 요소별로 MINDSTORMS 프로그래밍 학습의 효과를 알아보기 위해 하위 가설들을 t-검증한 결과는 다음과 같다.

가설 2 : MINDSTORMS 프로그래밍을 학습한 집단의 유창성, 융통성, 독창성, 정교성 점수는 학습하지 않은 집단보다 더 높을 것이다.

<표 5> 사후 창의력 검사에 대한 유창성 검사 비교

구 분	학생수	평균	표준편차	t	df	p
실험집단	30	46.23	10.59	.588	58	.558
통제집단	30	44.50	12.17			

*p < .05

첫째, 유창성에서는 MINDSTORMS 프로그래밍 학습여부에 따라서 유의한 차이가 없는 것으로 나타났다. <표 5>에 의하면, 사후 유창성 점수 평균은 실험집단이 46.23, 통제집단은 44.50로 나타났다. 또한 t 값이 0.588이고, 유의도가 0.558로 p<0.05 수준에서 두 집단은 통계적으로 유의한 차이가 크게 없는 것으로 나타났다.

<표 6> 사후 창의력 검사에 대한 융통성 검사 비교

구 분	학생수	평균	표준편차	t	df	p
실험집단	30	155.50	16.97	2.218	58	.030
통제집단	30	145.03	19.47			

*p < .05

둘째, 융통성에서는 MINDSTORMS 프로그래밍 학습여부에 따라서 유의한 차이가 있는 것으로 나타났다. <표 6>에 의하면, 사후 융통성 점수 평균은 실험집단이 155.50, 통제집단은 145.03로 나타났다. 또한 t 값이 2.218이고, 유의도가 0.030으로 p<0.05 수준에서 두 집단은 통계적으로 유의한 차이가 있었다. 따라서 MINDSTORMS 프로그래밍 학습이 학생들의 융통성 신장에 영향을 주었다고 할 수 있다.

<표 7> 사후 창의력 검사에 대한 독창성 검사 비교

구 분	학생수	평균	표준편차	t	df	p
실험집단	30	16.10	1.93	.140	58	.888
통제집단	30	16.03	1.73			

*p < .05

셋째, 독창성에서는 MINDSTORMS 프로그래밍 학습여부에 따라서 유의한 차이가 없는 것으로 나타났다. <표 7>에 의하면, 사후 독창성 점수 평균은 실험집단이 16.10, 통제집단은 16.03로 나타났다. 또한 t 값이 0.140이고, 유의도가 0.888로 p<0.05 수준에서 두 집단은 통계적으로 유의한 차이가 크게 없는 것으로 나타났다.

<표 8> 사후 창의력 검사에 대한 정교성 검사 비교

구 분	학생수	평균	표준편차	t	df	p
실험집단	30	32.56	4.34	2.327	58	.023
통제집단	30	29.63	5.36			

*p < .05

마지막으로, 정교성에서는 MINDSTORMS 프로그래밍 학습여부에 따라서 유의한 차이가 있는 것으로 나타났다. <표 8>에 의하면, 사후 정교성 점수 평균은 실험집단이 32.56, 통제집단은 29.63로 나타났다. 또한 t 값이 2.327이고, 유의도가

0.023으로 $p<0.05$ 수준에서 두 집단은 통계적으로 유의한 차이가 있었다. 따라서 MINDSTORMS 프로그래밍 학습이 학생들의 정교성 신장에 영향을 주었다고 할 수 있다.

이상의 결과를 정리해 보면 다음과 같다. MINDSTORMS 프로그래밍 학습을 한 집단이 학습하지 않은 집단보다 높은 창의력 신장을 보일 것이라는 가설에 대해서, 실험집단이 통제집단보다 높은 창의력 점수를 얻은 것으로 나타났고, 창의력 구성 요인별로는 융통성과 정교성의 사후검사 점수가 통제집단에 비해 실험집단에서 통계적으로 창의력 신장에 효과가 있는 것으로 나타났다. 모든 창의력 구성 요인별로 효과가 있는 것으로 나타나지는 않았지만, 전체적으로 MINDSTORMS 프로그래밍 학습을 한 집단이 학습하지 않은 집단보다 창의력 신장에 영향을 미치는 것으로 나타났다.

5. 결 론

기존 프로그래밍 교육은 내용과 방법, 대상에 대한 고려가 부족하였다. 특히, 초등학생의 발달 단계나 특성을 충분히 고려한 교육이 이루어지지 않아 교육현장에서 외면 받게 되었다. 이러한 문제를 해결하고자 본 연구에서는 주로 구체적 조작기에 있는 초등학생들의 발달 특성에 적합한 프로그래밍 교육 도구로서 MINDSTORMS의 활용을 제안하고 그 효과를 검증하고자 하였다.

MINDSTORMS을 이용한 프로그래밍 교육은 학습자들에게 로봇의 조립과 이를 동작시키기 위한 프로그래밍 과정에서, 직접적인 경험과 즉각적인 피드백을 제공함으로써 흥미와 이해도를 증진시킬 수 있으며, 프로그래밍 언어의 요소나 문제해결에 대해 쉽고 재미있게 학습할 수 있어 특히 초등학생들에게 효과적일 것으로 기대하였으며 특히 창의력 신장에 효과가 있을 것으로 보았다.

실험 결과, MINDSTORMS과 ROBOLAB을 활용한 프로그래밍 수업은 유의미한 결과를 보였다. MINDSTORMS 프로그래밍 학습을 한 집단이 학습하지 않은 집단보다 높은 창의력 신장을 보였으며, 창의력 구성 요인별로는 유창성, 독창성에서는 유의한 차이가 크게 없는 것으로 나타났고, 융통성, 정교성에 있어서는 실험집단이 통제집단보다 창의력 신장에 훨씬 효과가 있는 것으로 나타났다. 모든 창의력 구성 요인별로 효과가 있는 것으로 나타나지는 않았지만, 전체적으로 볼 때, MINDSTORMS 프로그래밍 학습이 창의력 신장에 영향을 미치는 것으로 나타난다.

프로그래밍 교육의 목적을 단지 프로그래밍 언어의 이해에 두는 것이 아니라 문제해결력, 논리적 사고력, 창의력 신장에 두고, 학생들이 스스로 그들의 학습활동을 통제하고 능동적으로 문제를 해결할 수 있는 환경을 조성해 주어야 하며, 이러한 프로그래밍 교육에 MINDSTORMS 은 매우 유용하고 적절한 도구가 될 수 있다. 또한, MINDSTORMS 프로그래밍 학습은 컴퓨터교육의 본질적인 목적인 창의적인 인간 육성을 위한 하나의 대안이 될 수 있을 것이다.

그리고, MINDSTORMS 프로그래밍 학습은 창의력 신장이라는 일차적인 목표 달성 외에도 다음과 같은 효과를 볼 수 있었다. 첫째, 학습자 스스로 즐겁게 능동적으로 학습에 참여한다. 둘째, 실제적, 구체적인 경험을 통해 학습하고 학습내용을 스스로 적용한다. 셋째, 자신의 생각을 창의적으로 표현한다. 넷째, 다양한 활동을 통해 논리·수학적 사고의 기초능력과 태도를 가진다. 끝으로, 다른 사람의 의견을 존중하고 협동할 수 있는 능력을 키워준다.

참 고 문 헌

[1] 김민정(2004). 동적기가환경(DGS)의 수학교육적 고찰-마이크로월드(MicroWorld)적 관점을 중심으로-. 서울대학교대학원 석학학위논문.

[2] 김재은(1991). 천재, 그 창조성의 비밀. 교보문고.

[3] 박종삼(1989). 비판적 사고와 창의성의 함양을 위한 교육과정 계획의 원리 탐색. 동국대학교 박사학위논문.

[4] 백영균, 우인상(1994). LOGO프로그래밍의 수업방식이 문제해결력에 미치는 효과에 관한 연구. 교육공학연구 9(1), 73-90.

[5] 서영목(1982). 가정환경요인과 창의성발달과의

관계. 고려대학교 석사학위논문.
[6] 이옥화(1993). 로고 프로그래밍의 교육적 의의와 실천 방안 모색. 교육공한 연구 8(1), pp.81-102.
[7] 이태욱(2000). 제7차 초·중등 컴퓨터 관련 교육과정의 분석 및 발전방안. 교원교육 16, pp.208-224.
[8] 정원식, 이영덕(1970). 표준화 간편 창의성 검사 실시 요강 및 규준(초등학교용). 서울: 코리안테스팅센터.
[9] 조미옥(1991). LOGO 프로그래밍의 안내적 교수법을 통한 인지적 모니터링 전략의 발달. 교육공학연구 7(1), pp.161-180.
[10] 조성연(1990). 아동의 창의성 발달 및 이에 관련된 생태학적 변인에 관한 연구. 연세대학교 박사학위논문.
[11] 한기정(1993). 유아의 창의성 교육을 위한 철학적·심리학적 기초. 단국대학교 박사학위논문.
[12] 허원호(1996). 인지양식에 따른 LOGO(꾀하미) 프로그래밍 학습이 창의력에 미치는 효과. 한양대학교 석사학위논문.
[13] Barry Fagin, Laurence Merkle (2003). Measuring the effectiveness of robots in teaching computer science. *ACM SIGCSE Bulletin , Proceedings of the 34th SIGCSE technical symposium on Computer science education, Volume 35* Issue 1, pp.307-311.
[14] Barry S. Fagin, Laurence D. Merkle, Thomas W. Eggers(2001). Teaching computer science with robotics using Ada/Mindstorms 2.0. *ACM SIGAda Ada Letters, Proceedings of the 2001 annual ACM SIGAda international conference on Ada, Volume XXI* Issue 4, pp.73-78.
[15] David J. Barnes(2002). Teaching introductory Java through LEGO MINDSTORMS models. *ACM SIGCSE Bulletin, Proceedings of the 33rd SIGCSE technical symposium on Computer science education, Volume 34* Issue 1, pp.147-151.
[16] Guilford, J. P.(1967). The nature of human intelligence. New York: McGrow-Hill.
[17] Holly Patterson-McNeill, Carol L. Binkerd(2001). Resources for using lego mindstorms. *Journal of Computing Sciences in Colleges, Volume 16* Issue 3, pp.48-55.
[18] LEGO Group. MINDSTORMS. http://www.LEGO.com/eng/education/MINDSTORMS/.
[19] Papert, S.(1980). MINDSTORMS: Children, computers, and powerful ideas. New York: Basic Books.
[20] The Free Encyclopedia Wikipedia: LEGO Mindstorms. http://en.wikipedia.org/wiki/Mindstorms
[21] Thomas R. Flowers, Karl A. Gossett(2002). Teaching problem solving, computing, and information technology with robots. *Journal of Computing Sciences in Colleges, Volume 17* Issue 6, pp.45-55.
[22] Torrance, E. P.(1974). Torrance Tests of Creative Thinking: Directions manual and scoring guide(Figual test booklet A, B). Scholastic Testing Service, Inc.

유 인 환

2000 한국교원대학교 컴퓨터교육과(교육학박사)
현재 대구교육대학교 조교수
E-Mail: bluenull@dnue.ac.kr

김 태 완

2005 대구교육대학교 컴퓨터교육과(교육학석사)
현재 대구효성초등학교 교사
E-Mail: kksan99@hanmail.net