알고리즘적 사고 문제 모델을 이용한 두리틀 프로그래밍 문제 개발 및 적용

Development and Analysis of Elementary Dolittle Programming Problems using Algorithmic Thinking-based Problem Model

허 경*

Kyeong Hur*

요 약

본 논문에서는 정보교육과정의 "문제해결방법과 절차" 영역에 필요한 알고리즘적 사고 문제 모델을 활용하고, 두리틀 프로그래밍 내용 요소를 적용하여 두리틀 프로그래밍 알고리즘적 사고 문제를 제안하였다. 그리고 개발된 두리틀 프로그래밍 알고리즘적 사고 문제들에 대해 실험수업을 실시하여 알고리즘적 사고에 따른 답안들의 다양성과 효율성 및 개발된 문제들의 난이도 적절성에 대한 분석을 통해 본 논문에서 제안된 문제들에 대한 두리틀 프로그래밍 알고리즘적 사고 문제로서의 적합성을 검증하였다.

Key Words : Computer Science Education, Educational Programming Language, Dolittle, Object-oriented Programming

ABSTRACT

This paper proposes elementary Dolittle programming problems using the algorithmic thinking-based problem model with material factors in the elementary Dolittle programming. And this paper proves the validity of developed Dolittle programming problems in defining them as algorithmic thinking-based problems through experiments. The experimental results are analyzed in views of variety and effectiveness evaluation of answer algorithms and suitability of allocating degrees of difficulties to the developed Dolittle programming problems.

Ⅰ. 서 론

2007년 2월에 개정 공시된 중등 '정보' 교과 교육과정 및 2006년 2월 개정된 ICT 운영지침에서는 정보화 사회에 필수적으로 필요한 알고리즘적 사고 기반의 문제해결능력을 키우기 위해 지금까지 정보 교과의 대부분을 이루었던 소프트웨어 활용 중심의 내용을 대폭 축소하고 컴퓨터과학의 원리에 대한 교육을 강화하였다[1-5].

문제해결 알고리즘적 사고는 Minsky(1967)가 정의한 "문제를 해결하는 효율적인 절차(A set of rules which tell us, from moment to moment, precisely

* 경인교육대학교 컴퓨터교육과 부교수 (khur@ginue.ac.kr)
제1저자 (First Author) : 허경
교신저자 (Corresponding Author) : 허경
접수일자 : 2011년 11월 21일
수정일자 : 2011년 12월 02일
확정일자 : 2011년 12월 10일

how to behave)"처럼 문제해결방법을 알고리즘으로 정의하는 사고이다. 이것은 Markov(1954)가 수학에서 정의한 알고리즘 "초기 자료로부터 원하는 결과로 이끌어주는 계산적 순서를 정의한 명확한 규정"과 유사하지만, 수학처럼 계산에 한정적인 것이 아니라 모든 문제를 대상으로 하고 있다.

위와 같은 알고리즘적 사고 기반의 문제해결력은 문제 풀이 과정을 다양한 형태의 알고리즘으로 만들어 보고 평가함으로써 향상될 수 있다. 따라서 알고리즘적 사고 기반의 문제해결력 향상을 위해서는 정해진 알고리즘을 가르치고 알고리즘의 이해도를 평가하는 방법보다 문제해결과정을 다양한 형태의 알고리즘으로 만들어 볼 수 있는 문제를 제시하고 학생 스스로 알고리즘을 만들어보고 그것을 평가할 수 있는 형태가 되어야 한다[5-8]. 기존의 알고리즘적 사고 학습을 위한 연구들을 살펴보면 대부분이 프로그래밍 과정에서의 알고리즘적 사고 학습을 목표로 하고 있으며, 알고리즘적 사고에 효과적인 프로그래밍 환경과 학습 내용을 함께 제시하고 있다[9-12].

개정된 정보교과 교육과정에서는 "문제 해결 방법과 절차" 영역에서 알고리즘적 사고 향상을 위해 "문제와 문제 해결 과정", "알고리즘의 개요", "알고리즘의 실체", "자료 찾기"와 같은 내용들을 제시하고 있다. 그러나 알고리즘적 사고를 직접적으로 체험해 볼 수 있는 "문제와 문제 해결 과정", "알고리즘의 개요", "알고리즘의 실체"와 같은 부분들은 구체적으로 내용을 제시하고 못하고 있다[2][3]. 지금 까지 우리나라에서 이루어졌던 프로그래밍 교육의 문제점을 정리해 보면, 첫째, 프로그래밍 교육을 위해서는 다소 많이 소요되는 시수확보가 어렵고 프로그래밍 강의를 위한 전문적인 교·강사 확보가 어렵다. 또한 표준화된 교육과정과 교재가 없기 때문에 지속적이고 체계적인 프로그래밍 교육이 이루어지지 못했다. 둘째, 프로그래밍 언어 자체의 어려움 즉, 다소 이해하기 힘든 문법, 복잡한 코딩내용, 프로그래밍 언어 사용방법의 어려움이 있어서 학습자들에게 부담을 안겨주게 되어 결국 흥미도와 성취도가 떨어지게 되므로 소기의 성과를 거두기가 힘들었다[13].

이에 본 논문에서는 정보교육과정의 "문제해결방법과 절차" 영역에 필요한 알고리즘적 사고 문제 모델을 활용하고, 두리틀 프로그래밍 내용 요소를 적용하여 두리틀프로그래밍 알고리즘적 사고 문제를 제안하였다. 그리고 제안된 두리틀 알고리즘적 사고 문제들에 대해 실험 수업을 실시하여 알고리즘적 사고에 따른 답안들의 다양성과 효율성, 개발된 문제들의 난이도 적절성에 대한 분석을 통해 본 논문에서 제안된 알고리즘적 사고 두리틀 EPL 문제 기반 언플러그드 (Unplugged) 교육이 컴퓨터를 많이 다룰 수 없는 지금과 같은 다양한 학교 현실에서도 적절한 효과를 보일 수 있음을 검증하였다.

II. 두리틀 프로그래밍 알고리즘적 사고 문제개발

1. 알고리즘적 사고 문제 모델

알고리즘적 사고란 컴퓨터가 수행할 수 있도록 문제해결방법을 정의하는 사고이다. 그러나 정보화 시대에는 모든 자료를 디지털형태로 처리하기 때문에 알고리즘적 사고는 단순히 컴퓨터에게 문제 해결 과정을 전달하기 위한 사고가 아니다. 본인의 문제를 다른 사람에게 정확하게 전달할 수 있는 사고이며 본인이 생각한 문제 해결 방법을 누가 수행해도 동일한 결과를 얻어 낼 수 있도록 정의하는 사고이다. 따라서 이러한 알고리즘적 사고를 학습하기 위한 문제는 특정 영역에 제한된 문제가 아니라 문제해결과정에 대한 알고리즘을 답으로 하는 문제이여야 한다. 또한 알고리즘적 사고 향상을 위한 문제는 문제해결과정이 다양하게 나올 수 있어야 하며 답으로 제시된 알고리즘에 대하여 평가 방법이 존재해야한다[12].

그림 1은 본 논문에서 적용하는 정보과학 알고리즘적 사고 문제 모델 이다[12]. 수학 문제와 달리 정보과학 알고리즘적 사고 문제는 주어진 도구(순서도 기호)를 조합하여 입력과 출력에 맞도록 블랙박스를 논리적으로 구성하는 것이다. 따라서 문제의 답안 내용은 블랙박스를 구성함으로써 문제해결방법을 만들어 내는 것이다. 즉, 문제 해결 방법을 이해하기보다 다양한 문제해결방법을 구성하는 데 중점을 두고자 하는 것이다.

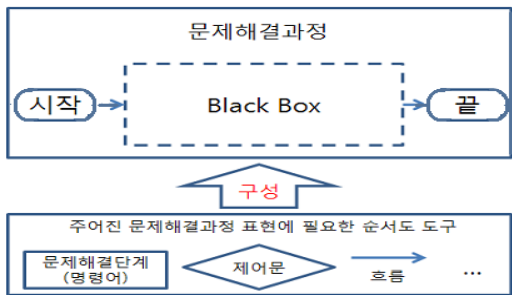


그림 1. 알고리즘적 사고 기반 문제 모델
Fig. 1. Algorithmic Thinking-based Problem Model

2. 두리틀 프로그래밍 언어

두리틀은 2001년 일본의 가네무네가 컴퓨터 교육을 위한 목적으로 개발한 객체지향형 교육용 프로그래밍 언어이다. 두리틀은 어린 학생들도 쉽게 이해할 수 있는 텍스트 기반의 교육용 프로그래밍 언어이며, 언어 설계에 있어 LOGO에서 거북 그래픽스의 아이디어와 인크리멘탈 프로그래밍 방식, 즉각적인 피드백 등 많은 교육적 아이디어를 수용하였다. 두리틀은 LOGO의 교육적 이점을 수용하면서 최근 프로그래밍 경향인 객체지향 패러다임에 부합되는 언어를 고안하기 위하여 Self 언어의 프로토타입 방식을 수용하여 어린 학생들이 어려운 객체지향 개념(클래스, 상속, 인스턴스 등)을 모르고도 쉽게 객체지향 프로그래밍을 체험할 수 있도록 고안한 언어이다[14].

2001년 11월 13일에 일본판 두리틀 정식 버전을 발표하였으며, 2003년 6월에는 초기 두리틀 1.0 버전의 인터페이스 및 명령어를 업그레이드하여 1.13 버전을 발표하였다. 그리고 2003년 8월부터 우리나라에서 처음으로 명령어 한글화 작업을 진행하였으며, 2003년 11월 8일 두리틀 한글 1.16 버전을 공식발표하였다. 그 후 1.19j 버전에서는 일본의 웹 서버로의 자료 업·로딩이 가능하고, 일본에서 작성한 소스코드가 한국 소스코드로 자동 변환되기 때문에 일본 학생과 한국 학생 간의 실시간 협동 프로젝트가 가능하며, 멜로디 기능도 첨가되어 학생들에게 컴퓨터 뿐만 아니라 다양한 영역의 탐구 학습을 체험할 수 있는 교육용 프로그램으로 개발되었고 현재는 2.10버전까지 나와있는 상태이며 계속 진행 중이다. 두리틀의 실행 화면은 그림 2와 같이 편집창과 실행창으로 구성되며, 우측의 편집버튼과 하단의 명령버튼으로 구성되어 있다.

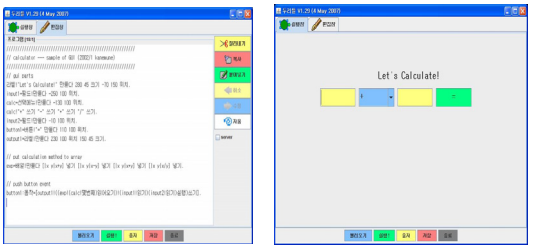


그림 2. 두리틀 편집창과 실행창
Fig. 2. Dolittle Program

프로그래밍 지도의 진정한 목표는 특정한 컴퓨터 언어의 문법 학습이 아니라 알고리즘의 이해와 표현 그리고 실행이다[15]. 두리틀은 다중 명령어 방식의 한글 프로그래밍 언어로 동일한 의미의 다중명령어 집합에서 학습자가 자신의 인지 수준에 맞는 명령어를 선택하여 사용할 수 있으며, 한글 어순의 간결한 문장으로 표현되기 때문에 어린 학생들도 쉽게 프로그래밍 할 수 있는 언어이다. 또한 기존의 타 언어에 비하여 함수적 표현이 용이하며, 거북 그래픽스환경으로 그래픽 표현도 용이하다. 두리틀은 컴퓨터교육을 목적으로 개발된 언어이지만 많은 수학적 요소를 포함하고 있으며, 두리틀 언어의 수식 표현이 수학에서의 수식 표현과 거의 유사하므로 변수와 함수의 개념학습에도 큰 도움을 줄 수 있다[14] 또한 바꾸어서 기존 수학교과에서 프로그래밍 학습 요소를 추출하여 프로그래밍 교육을 함에 있어서도 자연스러울 수 있다.

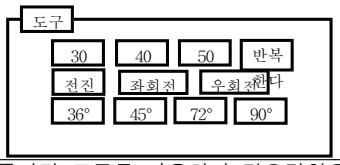
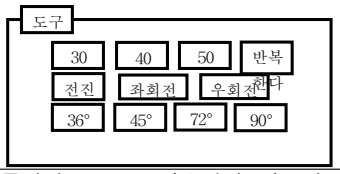
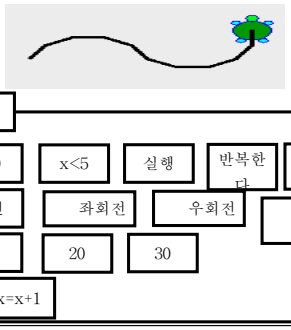
3 두리틀 프로그래밍 요소를 이용한 문제 개발

본 논문에서는 앞에서 정의한 알고리즘적 사고 기초문제 모델에 표 1의 두리틀 프로그래밍들 중 적용 가능한 요소들[14]을 이용하여 두리틀 프로그래밍 알고리즘적 사고 문제들을 표 2와 같이 개발하였다. 두리틀 프로그램은 소스가 오픈 되어 있는 특성상 버전에 따라 명령어가 약간씩 차이가 있어 교사와 학생이 동일한 버전을 사용함을 원칙으로 하였다. 각각의 문제는 표 2의 도구를 구성할 수 있는 형태로 제공하였다. 두리틀 프로그래밍의 내용 영역은 표 1과 같고 그중 문제를 만들기에 적합하지 않은 내용요소는 배제하였다. 예로 '수치' 영역은 그 내용이 아동의 교과학습 부분과 많은 차이가 있는 내용이 대부분을 이루고 있음으로 문제해결과정을 알고리즘 형태로 만들기에 적합하지 않아 문제 개발 영역 제외하였다.

표 1. 두리틀 프로그래밍 내용요소
Table 1. Dolittle Programming Factors

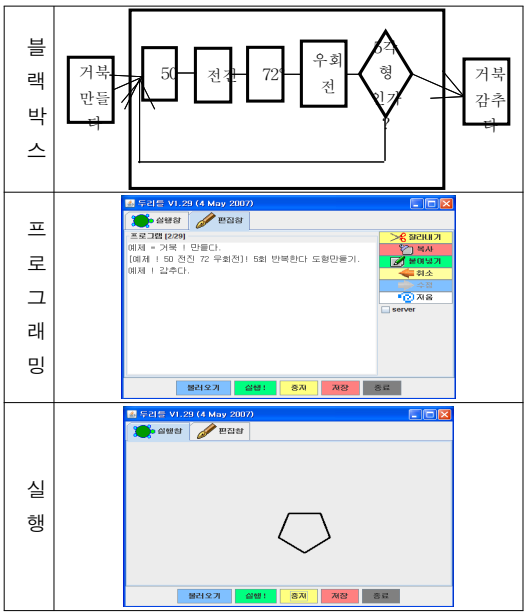
내용영역	내용요소
도형	만들다, 우회전, 좌회전, 이동, 위치한다, 색칠하다, 확대한다, 감추다, 보이다, 충돌
타이머	만들다, 간격, 시간, 횟수, 실행, 기다린다, 중단
수치	+, −, *, /, %, ==, !=, >, <, >=, <=, sqrt, sin, cos, tan, asin, acos, atan, round, ceil, floor, exp, log, pow, abs, random
문자열	비교연산, 실행, +, 연결한다, 포함한
블록	반복한다, 이라면, 그렇지않으면, 반복하는동안
버튼	만들다, 위치한다, 이동, 크기, 감추다, 보이다,
배열	만들다, 넣기, 읽어오기

표 2. 수학내용요소를 적용한 두리틀 EPL알고리즘적 사고 문제
Table 2. Algorithmic Thinking-based Dolittle Programming Problems

유형	문제 내용
순차형	문제) 주어진 도구를 이용하여 사각형을 만드는 방법을 구성하시오. 
	문제) 주어진 도구를 이용하여 정사각형을 만드는 방법을 구성하시오 
반복형	문제) 주어진 도구를 이용하여 거북이를 360도 회전 후 원래 위치로 돌아오도록 하는 방법을 구성하시오. 
	문제) 주어진 도구를 이용하여 정오각형을 만드는 방법을 구성하시오. 
분기형	문제) 주어진 도구를 이용하여 다음의 곡선을 그리는 방법을 구성하시오. 

학생들이 작성한 블랙박스에 사용한 도구를 바탕으로 두리틀 프로그래밍을 한다. 두리틀 프로그래밍은 비교반을 두어 문제해결에 대한 효과성을 검증한다. 비교반은 순서도 학습을 하지 않고 일반적 명령어 학습만을 하여 프로그래밍 하고 블랙박스를 사용하는 학생들은 블랙박스를 먼저 작성하여 알고리즘 학습을 한 후 다시 두리틀을 통한 프로그래밍을 한다. 이를 바탕으로 전통적 프로그래밍 교육을 받은 아동과의 차이를 통해 효과성을 검증한다. 표 2에서는 순차형, 반복형, 분기형의 3가지 3단계 형태만 보여지나, 혼합형 까지 4가지 4단계의 난이도로 구성될 수 있다. 표 3은 분기형의 예제 답안 및 두리틀 프로그램 화면의 모습이다. 표 3은 두리틀로 오각형을 만들기 위해서 블랙박스를 구성하고 프로그래밍하여 그 결과를 확인한 그림이다. 블랙박스의 도구는 전진, 우회전 등 두리틀에서 직접 사용하는 명령어 형태로 제시하여, <표 3>과 같이 구성해 볼 수 있고 프로그래밍 하여 확인할 수 있다. 학생들은 우선 블랙박스를 구성하기 위하여 다양한 알고리즘을 사고하여야 하며 1문항에 대하여 1차시(40분)정도의 사고하는 시간이 필요하다. 블랙박스를 구성하지 않는 비교반은 바로 프로그래밍 과정으로 들어가며 문제해결력에 대하여 블랙박스를 구성한 학생과 비교하였다.

표 3. 분기형 문제 정오각형 만들기 답안
Table 3. Algorithmic Thinking-based Dolittle Programming Conditional Problems



III. 적용 및 결과 분석

(1) 정답안의 다양성

본 논문에서는 실험을 위해 난이도별(순차형, 반복형, 분기형)로 각각 3문제씩 총 9문제를 선정하여 학교 4학년에 적용하였다. 학생 수는 총 A팀 30명이며 사전검사에 따라 모든 학생이 컴퓨터 프로그래밍을

다루어 보지 않은 학생이다. A팀 30명의 문제별(난이도별)로 작성한 답안 들을 살펴본 결과, 다양한 형태의 답안을 보였다. 보여진 답안은 모두가 정답을 맞추지는 않았다. 이는 학생들의 다양하고 창의적인 답안 작성을 보여준다. 하지만, 기존에 다루어 보지 않은 순서도에 대한 이해부족 및 시간부족 등으로 블랙박스 작성에 어려움이 있음도 보여져, 교사의 학습 방식과 대상 학년 선정에 대한 연구가 좀더 이루어져야 할 것이다. 답안을 맞춘 학생의 결과를 분석해보면 순차형, 반복형, 분기형의 세 가지 형태 실행결과 문제해결과 상관없이 순차형에서 분기형으로 난이도가 높아질수록 답안의 형태가 다양하게 나타났다.

학생 답안을 점수화 하여 그에 대한 총점을 비교하였더니 표 4와 같았다. 순서도의 정확성과 두리틀 프로그래밍에 대한 정확도를 수치화 하기 위하여 X = A(B + C)라는 수식을 사용하였다. 여기서 X는 총점으로 10점을 최대점수로 하고, A는 정답인지 아닌지를 보며 정답이면 1로 정답이 아니면 0으로 대입한다. B는 순서도 작성에 대한 점수로 최대 5점으로, C는 두리틀 프로그래밍에 대한 점수로 최대 5점으로 한다. 정답을 맞추지 못한 경우에는 A에 0이 대입되어 X는 0이므로 점수에 영향을 주지 못하므로 제외하였다. 난이도별 점수의 평균을 비교하면 순차형과 반복형은 1.2점 차이를 보였고, 반복형과 분기형은 1.58점 차이를 보였다. 난이도별 점수 비교로는 큰 차이를 보이지 않으나 실제계산에서 X=0인 학생의 점수는 난이도가 높아질 수록 많아져 정답률에서 큰 차이를 보였으며, 이는 그림 3과 같다.

표 4. 문제 해결에 대한 총점 비교
Table 4. Comparison of Total Score for Dolittle Programming Problems

	순차형	반복형	분기형
1번	8.52	7.76	7.12
2번	8.79	7.24	6.85
3번	9.01	7.72	6.99
평 균	8.77	7.57	6.99

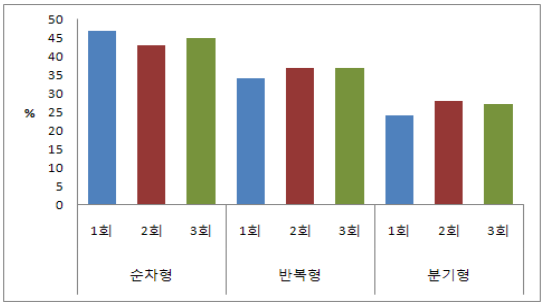


그림 3. 문제별 정답안의 비율
Fig. 3. Correctness ratio for each Problem

(2) 답안의 정확성 및 난이도의 적절성 분석

그림 3은 A팀 30명의 각각의 난이도별 3문제에 대한 답안의 정답률을 보여주는 그래프이다. 이 결과를 바탕으로 A팀의 블랙박스 적용 답안의 정확성을 보면 다음과 같다. 30명의 인원을 난이도별로 각각 3번씩 적용하여 정답을 맞춘 인원의 평균을 구해보았더니 순차형은 평균45%, 반복형은 35.5%, 분기형은 26.6%의 결과를 보였다. 이는 난이도가 순차형에서 분기형으로 높아짐에 따라 문제해결의 정확도가 낮아짐을 보여준다. 실험결과 중 특이한 점은 3차례 모두 정답이 맞은 아동은 없었으며, 문제마다 정답을 맞춘 학생이 서로 다르게 결과가 나와 교과학습능력이 크게 영향을 미치진 않는 것으로 분석되었다. 그림 6에서는 난이도에 따른 정답률의 차이를 비교할 수 있다. 난이도에 따라 정답률이 현저히 떨어지고 있음을 알 수 있다. 좀 더 다양한 문제제시와 난이도 조절이 필요하겠지만 어느 정도 난이도에 따라 정답률에 차이가 있음을 통해 난이도의 타당성을 보여준다.

Ⅳ. 결 론

기존의 연구들은 두리틀 프로그래밍의 교육에 대한 방법이나 프로그램 등의 개발을 주로 하였으나, 실제 학습한 내용을 활용하고 평가할 수 있는 평가문제에 대한 연구로서, 본 논문에서 개발된 알고리즘적 사고 기반 문제들에 대해 실험 수업을 실시한 결과로부터, 개발된 문제들은 여러 형태의 답안이 도출될 수 있는 알고리즘적 사고의 다양성을 지니며, 초등 단계에서 문제를 해결하는데 큰 어려움을 느끼지 않는 적절한 난이도를 지니고 있음을 확인하였다.

참 고 문 헌

[1] 교육인적자원부. 초중등학교 정보통신기술교육 운영지침, 2005.

[2] 교육인적자원부. 실과(기술·가정) 교육과정. 제 2007-79호 [별책 10], 2006.

[3] 교육인적자원부. 중학교 재량 활동의 선택 과목 교육과정. 고시 제 2007 - 79호 [별책 16], 2006.

[4] 한국교육학술정보원. 초중등학교 교육과정과 ICT통합안 연구. 한국교육학술정보원 연구보고 RM 2006-33. 2005.

[5] 이원규. 초중등학교 정보통신기술교육과 컴퓨터교육과정의 통합 방안 연구. 한국교육학술정보원 연구보고서 KR 2005-29. 2005.

[6] CSTA. ACM K-12 CS Model Curriculum. http://csta.acm.org/Curriculum/sub/k12final1022. pdf, 2003.

[7] Tim Bell, Ian H. Witten, Mike Fellows. Computer Science Unplugged, 2002

[8] France Belanger, Tracy Lewis, George M. Kasper, Wanda J. Smith, K. Vernard Harrington, "Are Computing Students Different? An Analysis of Coping Strategies and Emotional Intelligence," IEEE Transaction On Education , vol. 50, no. 3, pp. 188-196, AUG 2007.

[9] E. Milkova and M. Turcani. Digital Objects Supporting Development of Algorithmic Thinking. m-ICTE 2006 Current Developments in Technology-Assisted Education, Vol. 1, 2006.

[10] Stephen Cooper, Wanda Dann, Randy Pausch. Developing Algorithmic Thinking With Alice. ISECON 2000. Philadelphia, PA, November, 2000.

[11] Dagiene V. Algorithmic thinking: what set of Logo constructions should be used to develop it?, Proceedings of the 8th European Logo Conference EUROLOGO'01, Linz, 21-25 August, 2001, p. 245 - 252.

[12] 권대용 · 허경 · 이원규. (2008). 알고리즘적 사고 문제 모델 및 평가방법의 제안과 초등수학 내용 요소의 적용 및 분석. 컴퓨터교육학회 논문지, 11(4), 1-9.

[13] 김준영 · 문교식 (2009). 초등학교 정보통신 기술 교과서 프로그래밍 영역 내용 비교·분석. 정보교육학회논문지, 9권 P.151.

[14] 황우형 · 김경미 (2005). 객체지향형 교육용 프로그래밍 언어 '두리틀'의 수학교수-학습 활용 방안. 한국수학교육학회지.

[15] 조한혁 (1991). 교육용 언어의 설계에 대한 연구. 서울대학교 사대논총 제43호.

허 경 (Kyeong Hur) 정회원

1998년 : 고려대 전자공학사

2000년 : 고려대 전자공학석사

2004년 : 고려대 통신공학박사

2004년 8월~2005년 8월 : 삼성종합기술원(SAIT) 전문연구원

2005년 9월~ 현재 : 경인교대 컴퓨터교육과 부교수

<관심분야 > : 통신 및 네트워크, MAC, 컴퓨터교육