영상처리 소프트웨어 개발을 위한 비주얼 프로그래밍 환경

정 호 형* , 이 철 우* , 오 원 근**

*전남대학교 컴퓨터공학과 , **한국전자통신연구원

A Visual Programmig Environment for Image Processing Software Development

Jung Hohyoung* , Chil-Woo Lee* , Weon-Geun Oh**

*Department of Computer Engineering, Chonnam National University

, **Electronics and Telecommunications Research Institute

요 약

본 논문에서는 데이터 플로우 표현방식을 이용한 비주얼 프로그램으로 개발하려는 소프트웨어의 프로토타입을 빠르고 쉽게 만들어 완성된 소프트웨어를 시뮬레이션 할 수 있는 환경을 제공하는 TOI (Task Operation Integrator) 시스템을 소개한다. 이 시스템은 Liveness를 제공하여 사용자가 효과적으로 프로그램을 작성할 수 있게 해주고 다양한 환경에 융통성 있게 확장 가능하다.

1. 서론

영상처리 기술의 활용이 보편화되고 다양한 분야에 적용되어짐에 따라 더 많은 소프트웨어 개발자들이 영상처리 소프트웨어 개발에 참여하게 되었다. 이에 따라 기존의 중복되어 개발되던 기술들을 일반화하고 소프트웨어의 개발 기간을 단축시키기 위한 재사용 가능한 소프트웨어 라이브러리의 필요성이 대두되었고 그 결과로 여러 상용, 연구용 영상처리 소프트웨어 라이브러리[1]들이 제작되었으나 개발자가 이를 활용하기 위해서는 어느 정도의 프로그래밍 경력과 라이브러리 자체에 대한 이해가 필요하므로 간단히 자신의 아이디어를 실험하기 위한 프로그램 제작에도 적지 않은 시간이 소요되어 프로젝트의 수행을 더디게 하였다.

그래서 소프트웨어 라이브러리의 사용에 있어 보다 쉽고 편한 방법을 제공하려는 여러 연구들이 진행되어 왔는데 본 논문에서 소개하는 TOI (Task Operation Integrator) 시스템은 비주얼 프로그램의 형식을 이용하여 소프트웨어 라이브러리가 갖춘 유용한 명령들을 재활용하여 개발하고자 하는 소프트웨어의 프로토타입을 빠르고 쉽게 만들어 완성된 소프트웨어를 시뮬레이션 할 수 있는 환경을 제공하여 사용자와 라이브러리간의 격차를 줄여준다.

영상처리 소프트웨어는 전형적으로 일련의 처리를 영상 데이터에 가해 원하는 정보를 얻어내는 구조를 가지고 있어 마치 영상 데이터가 여러 처리과정을 따라 흘러가는 듯한 느낌을 주기 때문에 데이터 플로우(Data Flow)표현방식의 비주얼 프로그래밍으로 개발할 소프트웨어를 시뮬레이션하기에 적절하다.

TOI 시스템은 데이터 플로우 표현방식을 이용한 비주얼 프로그래밍 환경으로 사용자는 이를 이용하여 아이콘으로 표현되는 여러 처리 오퍼레이터(Task Operator)중에서 원하는 오퍼레이터를 선택하여 프로그래밍 영역에 위치시키고 이들을 서로 연결하여 데이터 플로우를 만듦으로서 완성된 소프트웨어의 프로토타입을 작성하고 시뮬레이션 할 수 있다.

사용자와의 효과적인 상호작용을 지원하기 위해 TOI 시스템은 오퍼레이터 개별 실행 방식을 이용하여 사용자가 오퍼레이터의 입력을 변화시킴과 동시에 별도의 실행 명령 없이도 새로운 입력 값에 영향을 받는 모든 오퍼레이터들을 올바른 실행 순서에 따라 차례로 수행되도록 하여 오퍼레이터의 자동성(Liveness)[3][4]을 실현하였다.

사용되는 분야가 다양하기 때문에 기존에 제공되는 처리 오퍼레이터들 외에 사용자가 원하는 기능의 처리 오퍼레이터를 추가해야 할 경우가 있게 마련이다. 이런 경우 사용자가 쉽게 새로운 오퍼레이터를 만들 수 있도록 하기 위해 사용자의 프로그래밍 능력에 따라 3가지 레벨의 기능 추가 모듈을 만들 수 있도록 하였다.

2. TOI 시스템

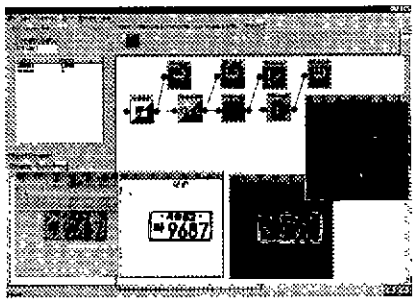

그림 1. TOI 시스템의 실행화면

2.1 오퍼레이터 (Operator)

사용자의 입장에서 오퍼레이터는 화면에 위치시킬 수 있는 일하는 기능을 수행하는 아이콘이다. 사용자는 이 아이콘들을 위치시키고 서로 연결하여 비주얼 프로그램을 만들고 이 프로그램은 원하는 실제 프로그램의 프로토타입이 될 수 있다. 사용자는 쉽게 프로토타입을 만들어서 실제 프로그램의 동작을 시뮬레이션 해보고 개선해 나가면서 시행착오를 줄일 수 있다.

개발자의 입장에서의 오퍼레이터는 OP Module 규약에 맞추어 만든 DLL 모듈이다. OP Module 규약은 소프트웨어 라이브러리에서 제공하는 기능을 오퍼레이터 DLL로 만들기 위해 어떻게 해야 하는지를 규정하고 있다. 개발자가 이 규약에 맞는 DLL을 개발하면 TOI 시스템에서 그 DLL을 이용할 수 있게 된다.

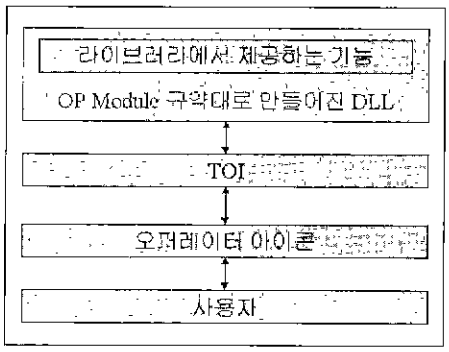

그림 2 오퍼레이터와 사용자

2.2 TOI 시스템에서의 비주얼 프로그래밍

TOI 시스템에서의 비주얼 프로그램은 데이터 플로우를 나타내는 네트워 모양을 하게된다. 사용자는 시스템에 등록된 여러 분야의 오퍼레이터들 중에 원하는 오퍼레이터들을 오퍼레이터 팔레트(그림 3)로부터 선택하여 위치시키고 서로 연결시킴으로써 오퍼레이터들 간의 데이터 전달을 표시하여 데이터가 연결된 순서대로 오퍼레이터들 자체는 모양을 만들게 된다. 이렇게 만들어진 데이터 플로우 네트워 위에 있는 오퍼레이터의 입력 값을 변화시킴으로써 실제 프로그램에서의 결과를 미리 예측해 볼 수 있다.

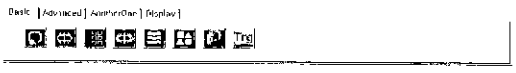

그림 3 오퍼레이터 팔레트

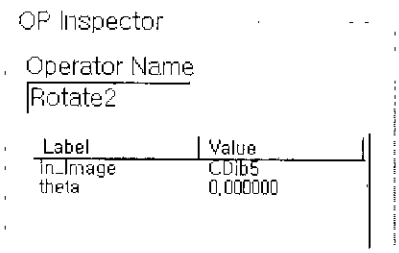

그림 4 오퍼레이터 인스펙터

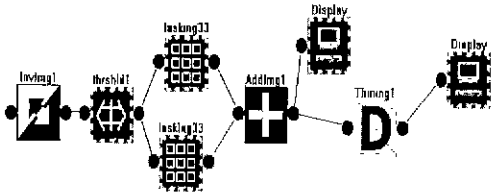

그림 5 데이터 플로우 네트워

3. 오퍼레이터의 자동성 (Liveness)

사용자와의 효과적인 상호작용을 돕기 위해 TOI 시스템은 오퍼레이터 개별 실행 방식을 이용해 오퍼레이터가 보다 빠르게 반응할 수 있게 하였다. 사용자가 임의의 오퍼레이터의 입력 데이터를 편집하게 되면 그 데이터를 입력으로 취하는 오퍼레이터가 실행되어 바로 변화된 값이 적용된 결과가 출력되게 하고 변화된 결과를 입력으로 하는 다른 오퍼레이터가 실행되어 이는 순간에도 프로그램은 최신의 정보를 유지하도록 한다.

이렇게 TOI 시스템은 오퍼레이터에 자동성을 주어 프로그램을 완전히 편집한 후 실행하는 기존의 인터프리팅 환경 보다 사용자가 시스템으로부터 빠른 반응을 얻어낼 수 있는 환경을 제공한다.

3.1 오퍼레이터와 데이터

오퍼레이터의 자동성(Liveness)은 알고리즘 수행함수인 오퍼레이터와 그 오퍼레이터가 실행되는데 필요한 입력 데이터, 알고리즘의 결과가 리턴되는 출력데이터의 상호작용으로 이루어진다.

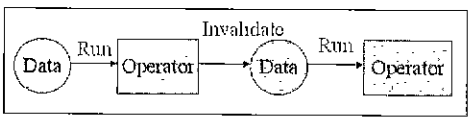

그림 6 오퍼레이터와 데이터의 상호작용

그림 6은 TOI 시스템에서의 데이터와 오퍼레이터가 자동성을 실현하는 방법을 간략히 보여준다. 오퍼레이터의 실행을 전적으로 관리하지 않고 오퍼레이터 자체에 위임하며 데이터들 역시 자신을 감조하는 오퍼레이터에 작접 접근할 수 있기 때문에 간단하지만 효율적인 실행 구조를 만들 수 있었다. 데이터는 자신의 상태가 변화되었음을 확인하게 되면 자신을 참조하고 있는 오퍼레이터에 실행(Run)메세지를 보내 오퍼레이터가 실행될 수 있게한다. 오퍼레이터는 자신이 수행된 후 출력 데이터들을 Invalidate 시킴으로써 그 데이터를 입력으로 참조하는 다음 오퍼레이터가 같은 동작을 수행하게 하여 데이터 변경이 일어나고 그 변경이 영향을 미치는 부분만을 올바른 순서로 수행되게 할 수 있다.

4. 시스템의 확장성

TOI 시스템은 크게 4가지의 구성 모듈로 나뉜다.

- TOI UI 및 여러 모듈 결합 (Executable)
- Operator Module : 순수 알고리즘 제공 (DLL)
- Type Module : 데이터의 생성, 삭제, 저장등을 담당 (DLL)

- Display Module 데이터의 디스플레이 방식을 제공 (DLL)

TOI 시스템에 Operator Type, Display Module의 형태를 갖추고 있는 DLL 모듈을 추가하여 확장할 수 있게 된다. 세 가지의 확장 모듈은 확장 모듈 개발자의 능력에 따라 기본적으로 C언어 프로그래이 가능한 개발자부터 윈도우즈 플랫폼에서의 프로그램 개발이 가능한 개발자까지 누구나 시스템을 확장할 수 있도록 한다

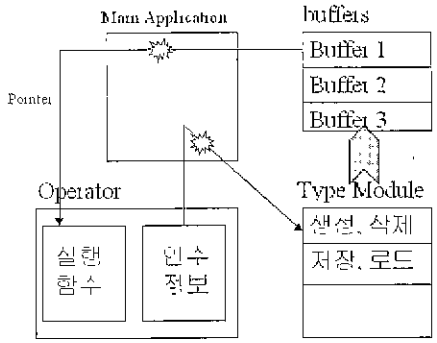

그림 7 모듈간의 상호작용

그림 7은 TOI 시스템에서의 오퍼레이터 모듈과 타입 모듈의 상호작용을 보여주고 있다. 데이터 버퍼의 생성과 삭제 등의 관리를 TOI 시스템에서 하게 된다. 이는 오퍼레이터가 순수 알고리즘만을 제공할 수 있게 해주며 알고리즘과 타입정보를 분리시켜 새로운 자료형의 추가를 용이하게 한다. 또한 오퍼레이터 계작에 있어서 일반 함수 작성과 동일한 방식을 제공해 준다

4.1 오퍼레이터 모듈 (Operator Module)

두 개의 외부함수 (Exported)함수[2]와 오퍼레이터를 나타낼 심볼 비트맵으로 구성된다. 구성이 비교적 간단하여 C언어 정도를 다룰 수 있는 수준이면 개발이 가능하다

- 알고리즘 수행함수 소프트웨어 라이브러리의 형식을 이용하는 알고리즘 수행함수
- 수행함수의 인수정보 제공함수 알고리즘 수행함수를 올바르게 호출할 수 있게 입출력 인수들의 타입, 이름 등의 정보를 제공 (TOI 시스템은 이 함수가 제공하는 인수정보를 이용하여 실제 수행함수를 호출하기 위한 작업을 함)
- 심볼 비트맵 프로그래밍 영역내에서 오퍼레이터를 표시할 비트맵

4.2 타입 모듈 (Type Module)

효율적이고 안정적인 시스템을 위해 모든 데이터 버퍼는 시스템에서 관리하게 된다. 그러나 시스템이 컴파일 될 당시 지원하지 않았던 자료형은 시스템이 생성, 삭제 등의 관리가 어렵다. 그런 이유로 시스템의 데이터 관리 능력을 확장해주기 위해 생겨난 모듈형식이다.

새로운 형의 데이터를 생성하거나 삭제해주고 저장이나 로드등의 기능을 기본적으로 제공하며 기본 형식만을 갖추고 있다면 Display Module과 상호 협력하기 위해 자유롭게 모듈에 새로운 기능을 추가할 수 있다

4.3 디스플레이 모듈 (Display Module)

시스템이 다루는 데이터는 주로 영상미디어인데 시스템에서 정해둔 디스플레이 방식만으로 충분히 데이터를 출력할 수 없을 경우가 있다. 이런 경우에 새로운 형태의 미디어를 출력 가능하게 만들기 위해 데이터의 타입 모듈과 상호 협조하여 화면에 출력할 수 있게 도와준다. 실질적으로 디스플레이 모듈은 타입모듈의 미디어 출력 함수를 이용하여 데이터를 출력할 수 있는 윈도우 클래스를 등록시키는 함수로 구성된다

5. 결론 및 향후 연구

앞에서 복잡한 영상처리 라이브러리를 사용자가 효과적으로 이용하게 만들어 주는 TOI 시스템을 소개하였다. 이 시스템은 다양한 환경에 융통성 있게 확장 가능하고 사용자에게 Liveness를 제공하여 보다 손쉬운 작업 환경을 제공하였다

그러나 아직 모든 확장 모듈들의 규약이 확실히 정해지지 않았고 여러 기지 상황에 적용해 보지 못했기 때문에 불완전하다

앞으로 다양한 응용분야에서 어떤 기능들을 더 필요로 하는지 그리고 그러한 기능을 제공하기 위해서 어떤 개선이 이루어져야 하는지를 연구할 필요가 있다

참 고 문 헌

[1] Khoral Research Inc, Khoros System Manual
[2] Jeffrey Richter, Advanced Windoes NT, Microsoft Press
[3] Does Continuous Visual Feedback Aid Debugging in Direct-Manipulation Programming Systems?, E M Wilcox, J W Atwood, M M Burnett
[4] Steering Programs via Time Travel, John Atwood, Margaret Burnett