영상처리 알고리즘 활용과 공유를 위한 효과적인 컴포넌트 구조에 관한 연구

어지수, 조영탁, 채옥삼
경희대학교 영상처리 연구실
{lafefish,greigen,oscahae}@khu.ac.kr

Efficient Sharable Component architecture for image processing and image processing algorithm development

Jeesoo Awe, Youngtak Cho, Oksam Chae
Kyunghee image processing lab.

요 약

전세계적으로 영상을 이용하는 분야는 급속도로 발전되고 확장되어 가고 있다. 이러한 시점에서 영상처리 기술의 중요성은 더욱더 강조 되고 있다. 하지만 영상처리 기술은 다른 기술들에 비하여 굉장히 특화되어 같은 기술이라도 사용 환경과 분야에 따라 각기 다른 형태와 다른 사용 방식을 요구하고 있다. 이러한 환경은 영상처리 기술들의 재사용성을 떨어뜨리고 기술 개발 비용의 증가를 가져 왔다. 본 논문에서는 이러한 문제점들을 해결하고자 영상처리 기술을 활용하고 공유하기 위한 컴포넌트의 구조를 제안하고자 한다.

1. 서 론

영상을 이용하는 기기와 컨텐츠의 일반화로 영상처리 기술이 제품 경쟁력에 중요한 부분을 차지하게 됨으로써 영상처리의 중요성은 더욱 강조되고 있다. 하지만 각 영상 처리가 사용되는 분야마다 특화된 알고리즘과 개발환경을 사용함으로써 알고리즘의 공유가 불가능해지고, 기본적인 기술 개발이 중복적으로 이루어짐으로써 연구 개발 비용이 증가하고 있다.

영상처리 분야	그 외 분야
같은 알고리즘이라도 적용하는 분야나 환경에 따라 다른 용도를 가진다.	같은 알고리즘은 비슷한 환경과 비슷한 분야에서 사용된다.
같은 알고리즘이라도 그 동작 조건에 따라 다양한 결과를 도출한다.	동작 조건에 따른 결과의 변화가 크지 않다
얻고자 하는 결과가 유사하더라도 그 결과를 위한 처리 과정이나 사용되는 알고리즘은 서로 다를 수 있다	얻고자 하는 결과가 유사하다면 그 처리 과정이 유사하다.
정확한 결과가 정해져 있지 않고 유사한 결과를 도출한다.	정확한 결과만이 가치가 있다.

[표 1 영상처리 분야의 특징]

이러한 문제점들은 표1의 영상처리의 특징들 때문에 일반적으로 이용되는 방법으로는 해결 할 수가 없다.

본 논문에서는 이러한 문제점들을 해결하고자 영상처리의 특화된 특징을 반영하여 공유 하기 쉬운 영상 처리 컴포넌트의 구조를 기존의 컴포넌트 방식인 COM과 기존의 영상 처리 알고리즘 개발 플랫폼인 Hello - Vision[3]을 이용한 해결 방안을 제시하고자 한다.

2. 관련 연구

기존의 영상처리 모듈개발을 위한 연구들은 영상처리 기능들을 컴포넌트화 하거나 자료형식을 표준화하려는 노력들이었다. 대표적인 것으로는 IUE, Khoros, PIKS, View-Station[3]이 있다. 위의 연구들이 다양한 방법으로 해결책을 제시해 주었지만 여러 가지 문제점[3]들을 여전히 내포하고 있기 때문에 영상처리의 특징들을 모두 반영하고 있지 못하다. 특히 위의 연구들은 모두 컴포넌트의 형식을 취하고 있지만 모두 개발 환경에 의존적이라는 단점을 가지고 있다.

관련 연구	문제점
IUE	자료 형식의 계층이 너무 깊고 복잡하다.

View - Station	확장이 어렵다.
Khoros & PIKS	다차원 배열과 튜플 구조 사용으로 인해 영상의 종류와 영상처리 결과를 적합하게 표현하지 못한다.

[표 2 기존 연구들의 문제점]

Hello - Vision[3]은 이러한 문제점들을 해결하고자 하는 목적의 일환으로 연구 되었으며 자료 형식을 간소화하고 확장성 있게 만들어진 환경이다. 하지만 Hello - Vision도 개발 환경에 의존적인 요소를 가지고 있다.

3. 영상처리 컴포넌트의 기본적인 조건

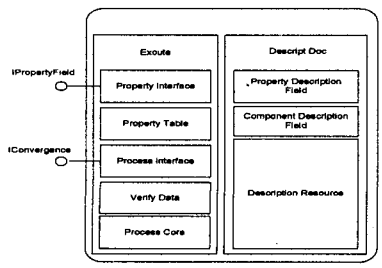
기본적으로 영상처리 컴포넌트는 [표1] 과 같은 영상처리 분야의 특징들을 반영 시켜 줄 수 있어야 하며, [표2]에서 볼 수 있는 문제점들을 해결 할 수 있어야 한다.

3.1 영상 처리 컴포넌트의 조건

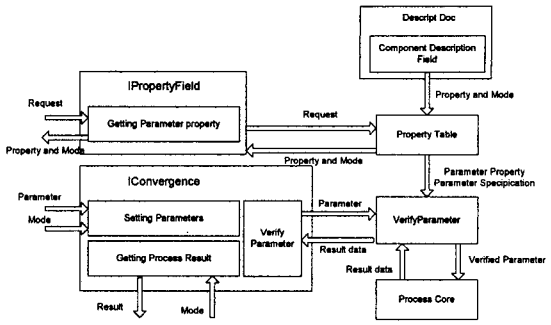
① 동작 조건을 조절 하기 쉬어야 함으로 파라메터 세팅이 자유로워야 한다.
② 사용되는 데이터 형식이 단순해야 한다.(공유 측면)
③ 영상을 표현하기 위한 자료 형식이 간결하고 사용하기 쉬어야 한다.
④ 사용 환경이나 Platform에 가능한 의존적이지 않아야 한다.
⑤ 컴포넌트끼리 조합이 쉬어야 한다.
⑥ 그 단위가 작아야 한다.(공유 측면)
⑦ 기술 정보를 영상처리에 맞는 형식으로 제공해 주어야 한다.(공유 측면)

위의 조건이 의미하는 가장 중요한 점은 영상처리의 특징들을 나타낼 수 있는 정보들과 다양한 분야의 사람들이 사용하기 편리한 컴포넌트의 구조와 정보를 갖추어야 한다는 것이다.

4. 영상처리 컴포넌트의 구조



[그림 1 영상 처리 컴포넌트의 구조]



[그림 2 영상 처리 컴포넌트의 상세 구조]

영상처리 컴포넌트에서 가장 중요한 것은 자유로운 이미지 데이터 처리와 동작 조건의 조작이다. 이를 위하여 (그림1)의 컴포넌트에서는 데이터를 주고 받는 부분의 데이터 형을 정하지 않고 각 데이터의 포인터만을 사용한다. 각 데이터의 형식은 미리 정의해 놓은 Property Description Field와 Component Description Field에서 가져와 사용하게 된다. 그 후 각 프로퍼티에 맞게 내부에서 데이터들을 처리 하게 됨으로써 각 모듈 마다 통일된 방식의 데이터 입출력 조작이 가능하게 되고, 다양한 모듈과도 결합 가능하게 된다

4.1 Interface

컴포넌트 Excute에서는 2개의 인터페이스만을 이용하여 정보를 주고 받게 된다. 2개의 인터페이스는 각각 데이터를 넣고 받을 수 있는 Mathod만을 가지며 Component Description Field에 정의한 Mode에 맞는 ID를 찾아서 데이터를 알맞은 곳에 전달 하고 가공하게 된다.

```
interface ICConvergence : IDispatch
{
HRESULT SetParameters([in]ULONG* pIMPCParam
                      ,[in]long nID);
HRESULT GetProcessingResult([in] int nMode ,
          [out, retval] ULONG* pIMPCReslt);};
```

SetParameters: 동작 조건과 영상 데이터를 받아 들인다.
GetProcessingResult: 처리된 결과 데이터를 노출 시킨다.

```
interface IPropertyField : IDispatch
{
HRESULT GetPropertyField([in] int nMode , [out,
retval] ULONG* pIMPCProp); };
```

GetPropertyField: 사용될 데이터들의 속성들을 Component Description Field에서 가져와 노출 시킨다.

4.2 Verify Data and Process core

Verify Data: 입력 받고 노출하는 데이터는 형태가 일반적인 포인터 타입이기 때문에 컴포넌트에서 실제로 사용될 형태로 전환하거나 내보낼 형태로 전환해 주는 역할을 한다. 데이터 형 변환은 컴포넌트 생성시 ID로 필터링하여 형 변환하며 이 형 변환된 자료들을 Process core로 전달해준다.

```
BOOL CVerifyData::VerifyParams(ParamateSet* pParamater)
{
        if(pParamater == NULL)
                return FALSE;
        switch(pParamater->m_nID)
        {
        case 0:
        sourceImage = (unsigned char*)pParamater->m_pData;
                break;
        ...
```

Process core: 실제로 영상처리 알고리즘이 구현되어 있는 곳이다. 여기서는 Hello Vision의 KSClass를 이용하여 구현하였다.

4.3 Description Doc

Property Description Field

영상처리 기술과 영상처리 알고리즘들에 대한 설명과 사용된 영상처리 모듈의 설명들이 기술 되어 있다.

Component Description

실제 컴포넌트 사용을 위한 기술적인 내용들이 들어가 있다. 실제로 이 부분을 참조하여 컴포넌트에서 사용된 자료형이나 데이터의 형태을 얻어오게 된다.

Description Resource

Property Description을 설명하고 표현하기 위한 영상자료나 아이콘등이 들어 있으며, Property Description Field의 기대 효과의 일부로써 처리 결과를 영상으로써 표현해 놓을 수 도 있다.

데이터 요소	정의
Title	제작자 나 발행자가 부여한 표제 제목
Creator	자원의 지적 내용에 책임을 진 인물이나 기관
Subject	주제나 그내용을 표현한 명사구나 구 사진을 설명하는 핵심어의 집합
Descrption	자원의 내용을 보다 자세하게 나타낸 텍스트 기술
Publisher	현재 형태로 제작된 자원의 제작 기관
Contributor	공헌자, 기고가
Type	자원의 범주나 장르 할당
Date	현재 형태의 자원이 제작된 연도
Format	자원의 데이터 표현 형식
Identifier	자원을 고유하게 식별하기 위한 문자열이나 기호
Source	해당 자원의 출처가 되는 저작
Language	자원의 내용을 기술한 언어
Relation	독립적으로 존재하는 자원간의 공식적인 관계
Coverage	자원에서 취급된 지역이나 시대나 장소
Right	권권이나 권권 관리 사항
Base Data	내용을 구성하기 위한 기반 데이터 요소
Basic Knowledge	내용을 구성하기 위한 기본적인 기반 지식 요소
Usage	단독으로 사용할시의 사용 용도등에 대한 리스팅 이나 예제
Expected Effect	다른 것들과 조립되어 사용될때 기대 효과나 예제

[표 3 Property Description Field]

component information	
Component type	컴포넌트의 종류
Interface	컴포넌트 interface 정보
interface Function	컴포넌트 Interface 함수 정보

[표 5 Component Description Field]

Interface info	
interface ID	인터 페이스 UUID아이디
interface purpose	인터 페이스 사용 목적

[표 6 Component Description Interface]

Type Table	
TypeID	타입 아이디, 모드
Type Real	실제 타입 정보

[표 7 Component Description Type Table]

5. 결론 및 향후과제

본 논문의 의의는 기존의 컴포넌트 형식을 이용한 영상처리용 컴포넌트를 개발 함으로써 그 동안 영상처리 분야의 특징이자 문제가 되었던 특화된 알고리즘의 공유가 힘들었던 환경을 극복하고 개발의 효율을 높이는 것에 둘 수 있다. 그러나 본 논문에서 제안된 구조는 아직 영상처리 특징들을 완전히 반영하고 효율적인 개발 환경을 제공하기 위한 일부분에 지나지 않는다.

향후 과제로는 지식 기반의 영상처리 컴포넌트 선정, 자료 형식의 표준화, 광범위한 영상처리 지식 및 알고리즘 공유를 위한 환경에 대한 끊임 없는 연구가 필요할 것으로 생각된다.

참고 문헌

[1] Geoge T. Heineman, William T. Councill , Component Based Software Engineering, Addison wessley 2001.

[2] Amy L. Lansky, Mark Friedman, Lise Getoor, Scott Schmidler, Nick Short Jr. The COLLAGE/Khros Link: Planning for Image Processing Task Recom Technologies/NASA Ames Resarch Center May 17 1995.

[3] 이정헌, 영상처리 기술의 체계적인 축적 및 활용을 위한 소프트웨어 컴포넌트 설계 및 재사용 환경 개발에 관한 연구 , 경희대학교, 2005. 2