10장 프레임 처리

- 프레임 처리의 개념
- 프레임 결합 처리
- 프레임 합성 처리
- 동영상의 개념

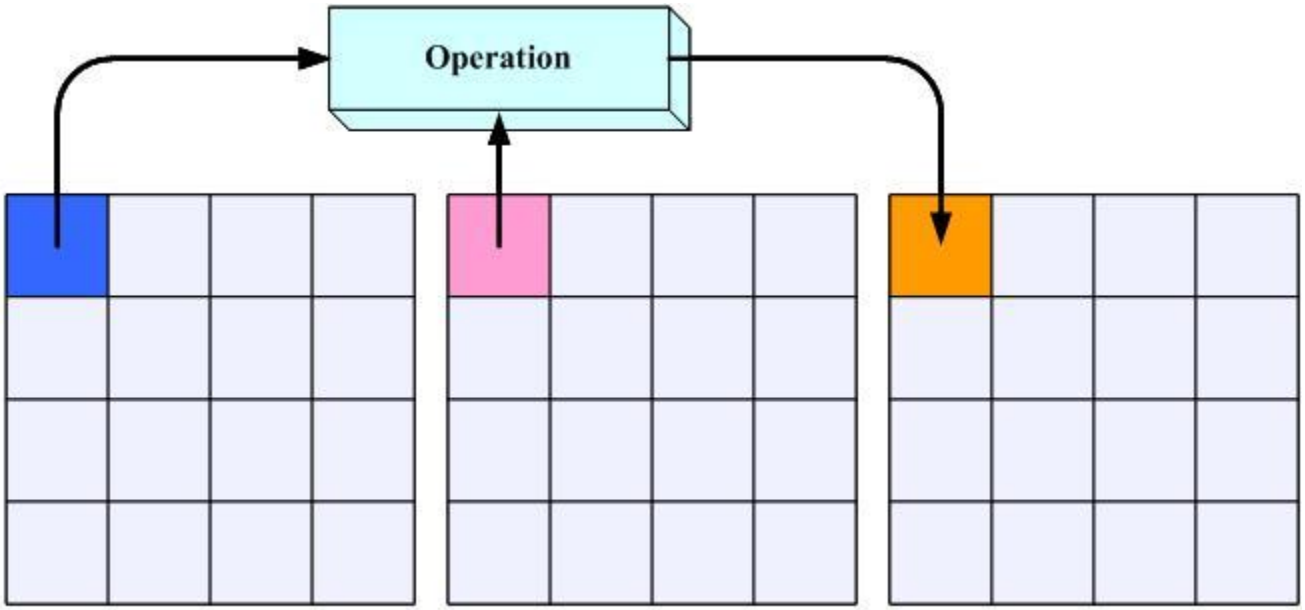
한빛미디어(주)

학습목표

- ✓ 프레임 처리의 개념을 이해한다.
- ✓ 결합 처리에서 사용되는 산술연산을 공부한다.
- ✓ 합성 처리에서 자주 사용되는 논리연산을 학습한다.
- ✓ 동영상의 구조를 이해하고 프레임 처리의 응용을 소개한다.

Section 01 프레임 처리의 개념

- 프레임 처리(Frame Processing)
  - 두 개 이상의 서로 다른 영상을 포함한 영상 간의 연산을 바탕으로 새로운 화소 값을 생성하는 것
  - 생성된 결과 영상의 각 화소는 입력 영상과 같은 위치에 생성됨.



[그림 10-1] 프레임 처리의 개념

  - 결합 처리(Combination Processing)와 합성 처리(Composition Processing)로 나뉨

프레임 결합 처리(계속)

- 결합 처리
  - 서로 관련 있는 복수의 영상을 합성하여 향상된 품질의 영상을 만드는 것으로 주로 산술연산으로 수행
  - 영상 간의 덧셈, 뺄셈, 평균연산 등이 대표적

$$O(x, y) = I1(x, y) + I2(x, y)$$

입력영상 1 입력영상 2 결과영상

[그림 10-2] 덧셈연산을 이용해 프레임 처리한 결과 영상

Section 02 프레임 결합 처리-덧셈 연산을 이용한 프레임 처리

- 덧셈 연산을 이용한 프레임 처리
  - 덧셈연산을 이용하여 프레임 처리한 결과 영상이 너무 밝아지는 것
  - 오버플로를 방지하기 위해 프레임 처리에서 α값을 이용한 공식

$$O(x, y) = \alpha I1(x, y) + (1-\alpha)\, I2(x, y)$$

- **α는 0~1 사이의 값**

입력영상 1 입력영상 2 결과영상

[그림 10-2] 덧셈연산을 이용해 프레임 처리한 결과 영상

[실습하기 10-1] 덧셈을 이용한 프레임 처리 프로그램

① **ResourceView** 창에서 **[Menu]-[IDR_IMAGETYPE]** 더블클릭 → 메뉴 추가

ID	ID_FRAME_SUM
Caption	프레임 덧셈

② **[MFC ClassWizard]** 대화상자를 이용해 추가된 메뉴에서 프레임 단위 덧셈을 실행하는 함수 추가

Class Name	Function Type	Function Name
View Class	void	OnFrameSum
Doc Class	void	OnFrameSum

③ **Doc** 클래스에 다음 프로그램 추가

[실습하기 10-1] 덧셈을 이용한 프레임 처리 프로그램

```
void CImageProcessingDoc::OnFrameSum()
{
    CFile File;
    CFileDialog OpenDlg(TRUE);

    int i;
    unsigned char *temp;

    m_Re_height = m_height;
    m_Re_width = m_width;
    m_Re_size = m_Re_height * m_Re_width;

    m_OutputImage = new unsigned char[m_Re_size];

    if(OpenDlg.DoModal() == IDOK){
        File.Open(OpenDlg.GetFileName(), CFile::modeRead);
        // 덧셈연산을 수행할 새로운 영상을 얻기 위해
        // 열기 대화상자를 이용해 영상을 입력
```

[실습하기 10-1] 덧셈을 이용한 프레임 처리 프로그램

```
        if(File.GetLength() == (unsigned)m_width * m_height){
            temp = new unsigned char[m_size];
            // 입력 값 저장을 위한 배열 선언

            File.Read(temp, m_size); // 선택된 파일을 읽어 배열에 저장
            File.Close();

            // 프레임 간에 픽셀 대 픽셀로 덧셈연산 실행
            for(i=0 ; i<m_size ; i++){
                if(m_InputImage[i] + temp[i] > 255)
                    m_OutputImage[i] = 255;
                else
                    m_OutputImage[i] = m_InputImage[i] + temp[i];
            }
        }
        else{
            AfxMessageBox("Image size not matched");
            //영상의 크기가 다를 때는 처리하지 않음
        return;
        }
    }
}
```

[실습하기 10-1] 덧셈을 이용한 프레임 처리 프로그램

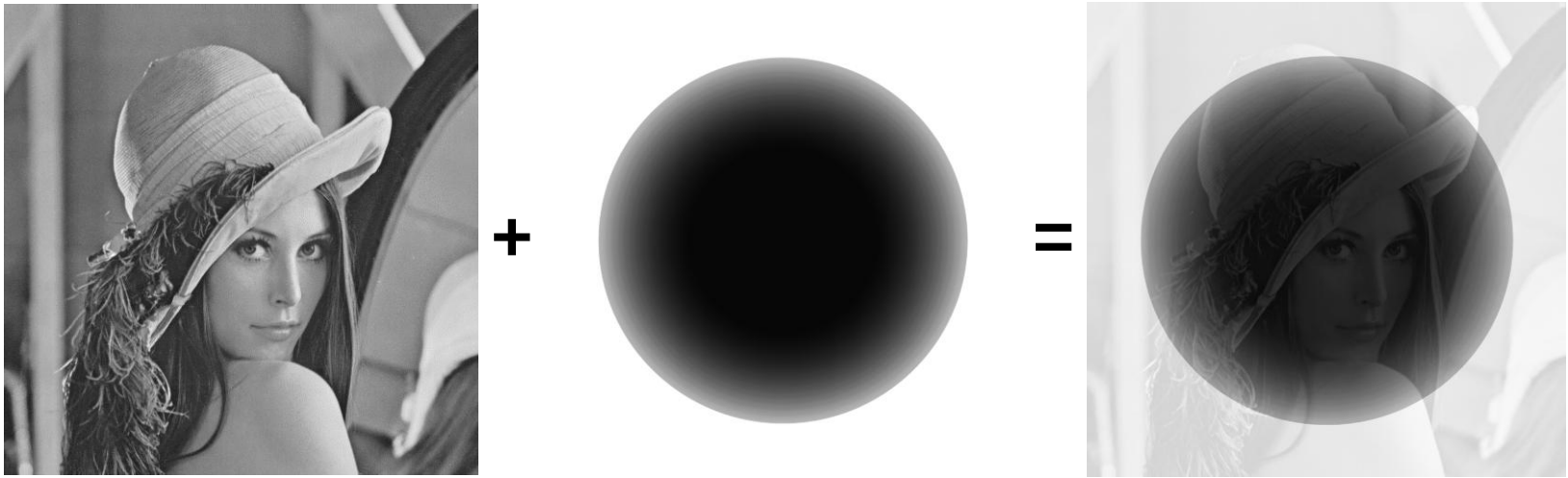
④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnFrameSum()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameSum();

    Invalidate(TRUE);
}
```

[실습하기 10-1] 덧셈을 이용한 프레임 처리 프로그램

⑤ 프로그램 실행 결과 영상

- 밝기가 그대로 유지되면서 덧셈 프레임 처리가 수행된 결과 영상임
- 덧셈연산은 그림에서 보는 바와 같이 특정 부분을 강조 및 제거하거나 가장 쉽고 간단하게 두 영상을 합성하는 데 응용 가능



입력영상 **1** + 입력영상 **2** = 결과영상

밝기가 같은 덧셈연산을 이용해 프레임 처리한 결과 영상

뺄셈연산을 이용한 프레임 처리

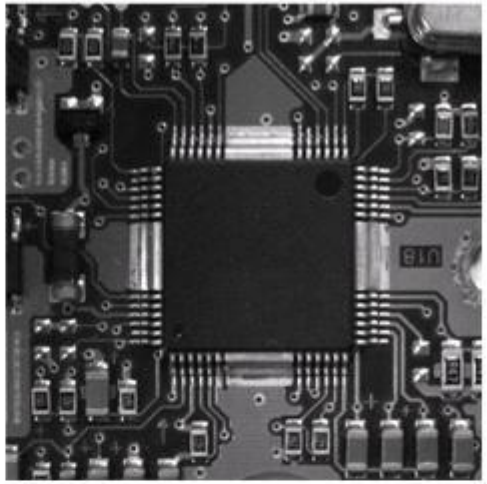
- 한 영상에서 다른 영상의 값을 빼서 두 영상 사이의 차이를 결정하는 프레임 처리
- 영상의 변화를 검출하는 데 효율적
- 뺄셈 프레임 처리 공식

$$O(x, y) = I1(x, y) - I2(x, y)$$

- 똑같은 장면을 다른 시간대에서 촬영해 얻은 영상을 뺄셈 처리하여 대상체를 식별하는 방법으로, 배경제거, 감시 시스템, 조립 라인의 검사 시스템, 불필요하게 추가되는 잡음제거 등을 응용하는 데 사용됨.

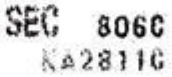
뺄셈연산을 이용한 프레임 처리(계속)

(a) 정상적인 영상



(b) 오류 영상

(c) 차 영상



(d) 차 영상의 이진화 영상

[그림 10-3] 뺄셈 프레임 처리를 이용한 검사 시스템 결과 영상

[실습하기 10-2] 빨셈을 이용한 프레임 처리 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_FRAME_SUB
Caption	프레임 빨셈

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 단위 빨셈연산을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnFrameSub
Doc Class	void	OnFrameSub

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 10-2] 뺄셈을 이용한 프레임 처리 프로그램

```
void CImageProcessingDoc::OnFrameSub()
{
   CFile File;
   CFileDialog OpenDlg(TRUE);

   int i;
   unsigned char *temp;

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char[m_Re_size];

   if(OpenDlg.DoModal() == IDOK){
      File.Open(OpenDlg.GetFileName(), CFile::modeRead);
```

[실습하기 10-2] 뺄셈을 이용한 프레임 처리 프로그램

```
        if(File.GetLength() == (unsigned)m_width * m_height){
            temp = new unsigned char[m_size];

            File.Read(temp, m_size);
            File.Close();

            // 프레임 간에 픽셀 대 픽셀로 뺄셈연산 실행
            for(i=0 ; i<m_size ; i++){
                if(m_InputImage[i] - temp[i] < 0)
                    m_OutputImage[i] = 0;
                else
                    m_OutputImage[i] = m_InputImage[i] - temp[i];
            }
        }
        else{
            AfxMessageBox("Image size not matched");
            return;
        }
    }
}
```

[실습하기 10-2] 뺄셈을 이용한 프레임 처리 프로그램

④ View 클래스에 다음 프로그램 추가

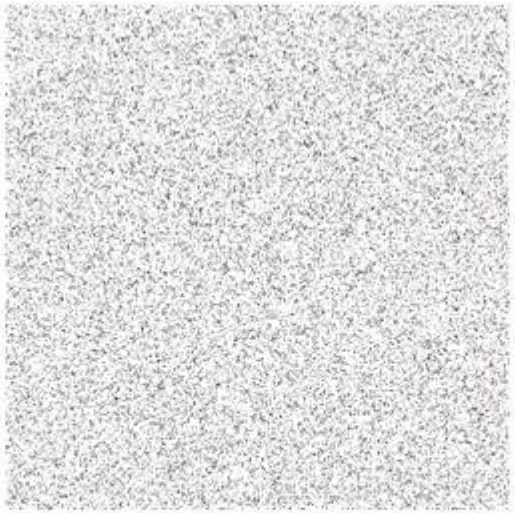
```
void CImageProcessingView::OnFrameSub()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameSub();

    Invalidate(TRUE);
}
```

[실습하기 10-2] 뺄셈을 이용한 프레임 처리 프로그램

⑤ 프로그램 실행 결과 영상

- 뺄셈 프레임 처리는 이미 아는 불필요한 잡음을 제거하는 데 사용 가능
- (a)는 잡음이 첨가된 입력 영상이고, 이 잡음은 이미 알려진 형태
- 입력 영상에서 알려진 잡음을 뺄셈하면 잡음이 제거됨.

입력 영상



잡음

잡음제거 영상

잡음 형태를 알 때 차 영상을 이용하여 잡음제거

곱셈연산을 이용한 프레임 처리

- 서로 다른 두 영상을 곱하여 새로운 결과 영상을 얻는 프레임 처리 방법
- 덧셈처럼 영상을 합성하는 역할 수행
- 곱셈연산을 이용한 프레임 처리 공식

$$O(x, y) = I1(x, y) \times I2(x, y)$$

[실습하기 10-3] 곱셈을 이용한 프레임 처리 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_FRAME_MUL
Caption	프레임 곱셈

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 양선형 보간법을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnFrameMul
Doc Class	void	OnFrameMul

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 10-3] 곱셈을 이용한 프레임 처리 프로그램

```
void CImageProcessingDoc::OnFrameMul()
{
    CFile File;
    CFileDialog OpenDlg(TRUE);

    int i;
    unsigned char *temp;

    m_Re_height = m_height;
    m_Re_width = m_width;
    m_Re_size = m_Re_height * m_Re_width;

    m_OutputImage = new unsigned char[m_Re_size];
```

[실습하기 10-3] 곱셈을 이용한 프레임 처리 프로그램

```
    if(OpenDlg.DoModal() == IDOK){
        File.Open(OpenDlg.GetFileName(), CFile::modeRead);
        if(File.GetLength() == (unsigned)m_width * m_height){
            temp = new unsigned char[m_size];
            File.Read(temp, m_size);
            File.Close();

            // 프레임 간에 픽셀 대 픽셀로 곱셈연산 실행
            for(i=0 ; i<m_size ; i++){
                if(m_InputImage[i] * temp[i] > 255)
                    m_OutputImage[i] = 255;
                else
                    m_OutputImage[i] = m_InputImage[i] *temp[i];
            }
        }
        else{
            AfxMessageBox("Image size not matched");
            return;
        }
    }
}
```

[실습하기 10-3] 곱셈을 이용한 프레임 처리 프로그램

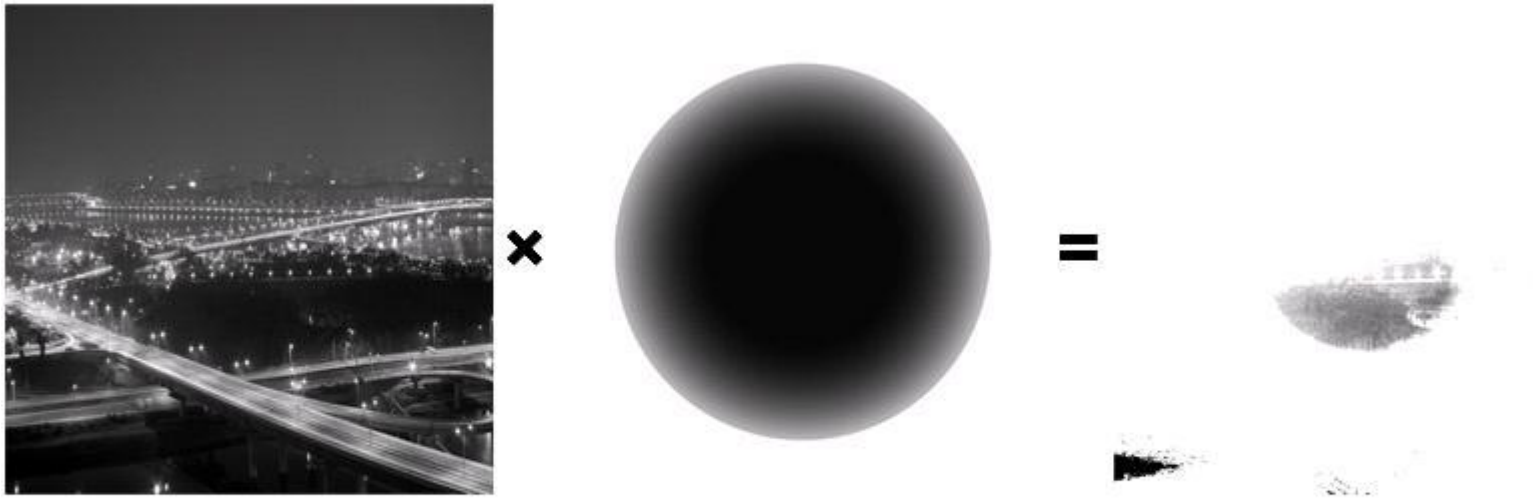
④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnFrameMul()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameMul();

    Invalidate(TRUE);
}
```

[실습하기 10-3] 곱셈을 이용한 프레임 처리 프로그램

⑤ 프로그램 실행 결과 영상

- 결과 영상이 뒤에 소개될 AND 연산과 효과가 같음.



곱셈 프레임 처리를 이용한 결과 영상

나눗셈연산을 이용한 프레임 처리

- 하나의 영상을 다른 영상으로 나누어서 새로운 결과 영상을 얻는 프레임 처리 방법
- 뺄셈연산과 비슷하게 검출되거나 밝기를 조절하는 역할 수행
- 나눗셈연산을 이용한 프레임 처리 공식

$$O(x, y) = I1(x, y) \ \ I2(x, y)$$

- 스펙트럼 비율 기법 **(Spectrum Ratio Scheme)**
  - 나눗셈연산을 이용한 대표적인 방법으로 살아 있는 식물과 죽은 식물을 구별할 때 유용

[실습하기 10-4] 나눗셈을 이용한 프레임 처리 프로그램

① **[MFC ClassWizard]** 대화상자를 이용해 추가된 메뉴에서 히스토그램 스트레칭을 실행하는 함수 추가

ID	ID_FRAME_DIV
Caption	프레임 나눗셈

② **[MFC ClassWizard]** 대화상자를 이용해 추가된 메뉴에서 프레임 단위 나눗셈 연산을 실행하는 함수 추가

Class Name	Function Type	Function Name
View Class	void	OnFrameDiv
Doc Class	void	OnFrameDiv

③ **Doc** 클래스에 다음 프로그램 추가

[실습하기 10-4] 나눗셈을 이용한 프레임 처리 프로그램

```
void CImageProcessingDoc::OnFrameDiv()
{
   CFile File;
   CFileDialog OpenDlg(TRUE);

   int i;
   unsigned char *temp;

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char[m_Re_size];

   if(OpenDlg.DoModal() == IDOK){
      File.Open(OpenDlg.GetFileName(), CFile::modeRead);

      if(File.GetLength() == (unsigned)m_width * m_height){
      temp = new unsigned char[m_size];

      File.Read(temp, m_size);
      File.Close();
```

[실습하기 10-4] 나눗셈을 이용한 프레임 처리 프로그램

```
        // 프레임 간에 픽셀 대 픽셀로 덧셈연산 실행
        for(i=0 ; i<m_size ; i++){
            if(m_InputImage[i] == 0)
            // 나뉘는 값이 '0'이면 출력은 영상에서의 최소값
                m_OutputImage[i] = 0;
            else if(temp[i] == 0)
            // 나누는 값이 '0'이면 출력은 영상에서의 최대값
                m_OutputImage[i] = 255;
            else
                m_OutputImage[i]
                    = (unsigned char)(m_InputImage[i] / temp[i]);
            }
        }
        else{
            AfxMessageBox("Image size not matched");
            return;
        }
    }
}
```

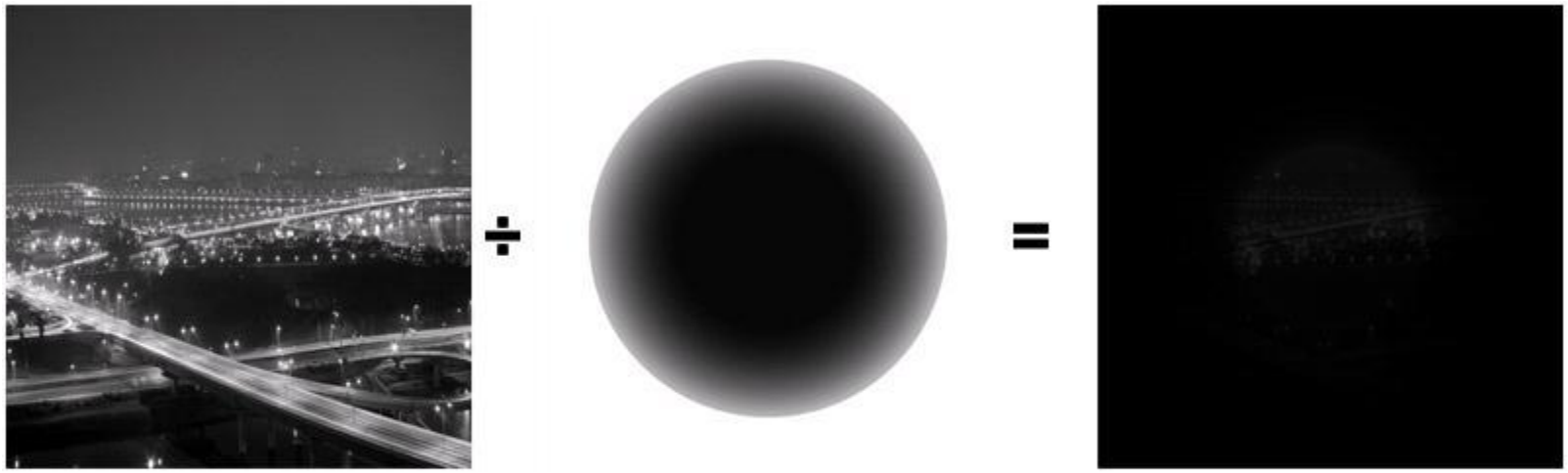
[실습하기 10-4] 나눗셈을 이용한 프레임 처리 프로그램

④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnFrameDiv()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameDiv();

    Invalidate(TRUE);
}
```

[실습하기 10-4] 나눗셈을 이용한 프레임 처리 프로그램

⑤ 프로그램 실행 결과 영상



나눗셈 프레임 처리를 이용한 결과 영상

평균연산을 이용한 프레임 처리

- 같은 영상 여러 장을 다른 종류의 잡음으로 훼손하였다면 모두 영상의 평균을 구해서 자연스럽게 잡음을 제거하면 됨.
- 특정한 상황에서 잡음을 제거하는 방법으로 사용됨.
- 특정한 상황은 영상을 전송할 때 나타남.
- 전송 중에 생성되는 각기 다른 잡음의 형태를 평균화하여 제거 가능
- 전송 영상 두 개의 평균을 구하는 공식

$$O(x, y) = \frac{I_1(x, y) + I_2(x, y)}{2}$$

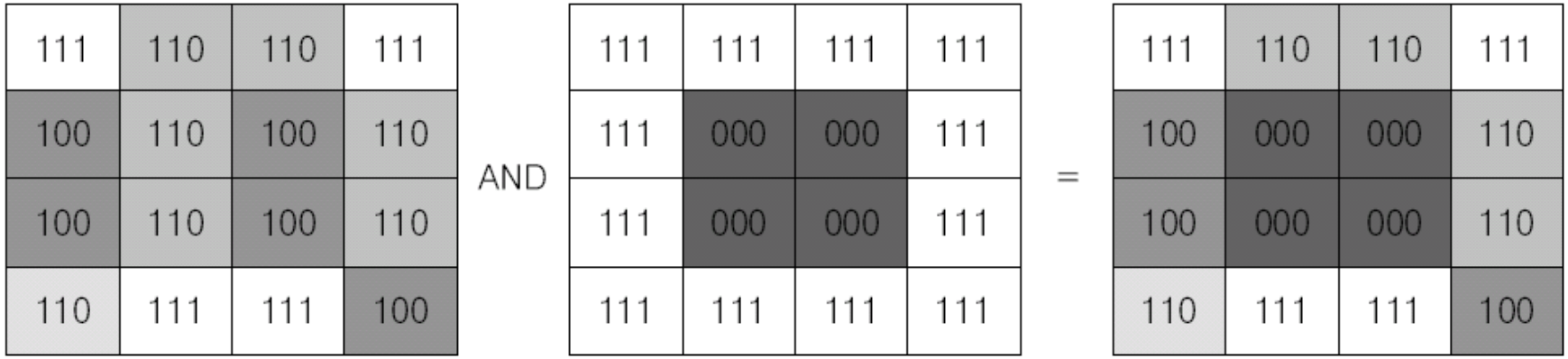
원영상 원영상 + 잡음 평균한 영상

[그림 10-4] 평균 프레임 처리를 이용한 결과 영상

Section 03 프레임 합성 처리-AND 연산을 이용한 프레임 처리

- **AND 연산을 이용한 프레임 처리**
  - 두 입력 데이터가 모두 1일 때만 1을 출력하고 나머지 세 종류의 입력에서는 모두 0을 출력하는 것
  - 영상의 특정 부분을 상쇄하는 데 사용
  - 영상의 합성 과정에서 AND 연산으로 상쇄된 부분에는 새로운 영상을 추가
  - **AND 연산을 이용한 프레임 처리 공식**

    $O(x, y) = I1(x, y)\ OR\ I2(x, y)$



[그림 10-5] AND 연산을 이용한 프레임 처리의 특징

[실습하기 10-5] AND 연산을 이용한 프레임 처리 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_FRAME_AND
Caption	프레임 AND

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 프레임 단위 OR 연산을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnFrameAnd
Doc Class	void	OnFrameAnd

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 10-5] AND 연산을 이용한 프레임 처리 프로그램

```
void CImageProcessingDoc::OnFrameAnd()
{
    CFile File;
    CFileDialog OpenDlg(TRUE);
    int i;
    unsigned char *temp;

    m_Re_height = m_height;
    m_Re_width = m_width;
    m_Re_size = m_Re_height * m_Re_width;

    m_OutputImage = new unsigned char[m_Re_size];

    if(OpenDlg.DoModal() == IDOK){
        File.Open(OpenDlg.GetFileName(), CFile::modeRead);
```

[실습하기 10-5] AND 연산을 이용한 프레임 처리 프로그램

```
        if(File.GetLength() == (unsigned)m_width * m_height){
            temp = new unsigned char[m_size];
            File.Read(temp, m_size);
            File.Close();

            // 프레임 간에 픽셀 대 픽셀로 AND 연산 실행
            for(i=0 ; i<m_size ; i++){
                m_OutputImage[i]
                    = (unsigned char)(m_InputImage[i] & temp[i]);
            }
        }
        else{
            AfxMessageBox("Image size not matched");
            return;
        }
    }
}
```

[실습하기 10-5] AND 연산을 이용한 프레임 처리 프로그램

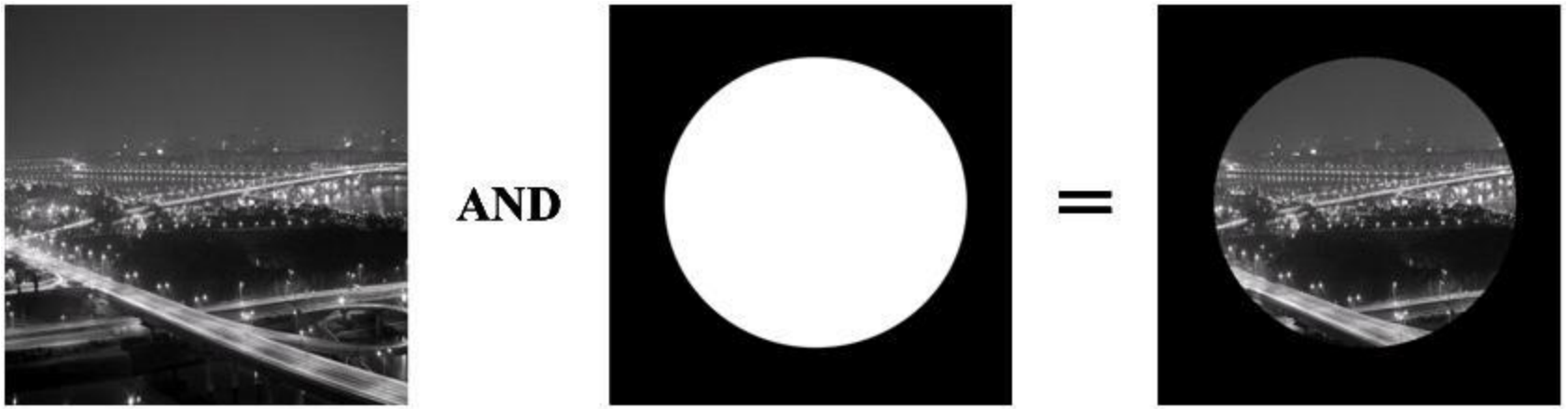
④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnFrameAnd()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameAnd();

    Invalidate(TRUE);
}
```

[실습하기 10-5] AND 연산을 이용한 프레임 처리 프로그램

⑤ 프로그램 실행 결과 영상

- 실제 영상으로AND 연산을 수행한 것으로, 특정 부분만 상쇄되고 원하는 부분은 그대로 유지됨.

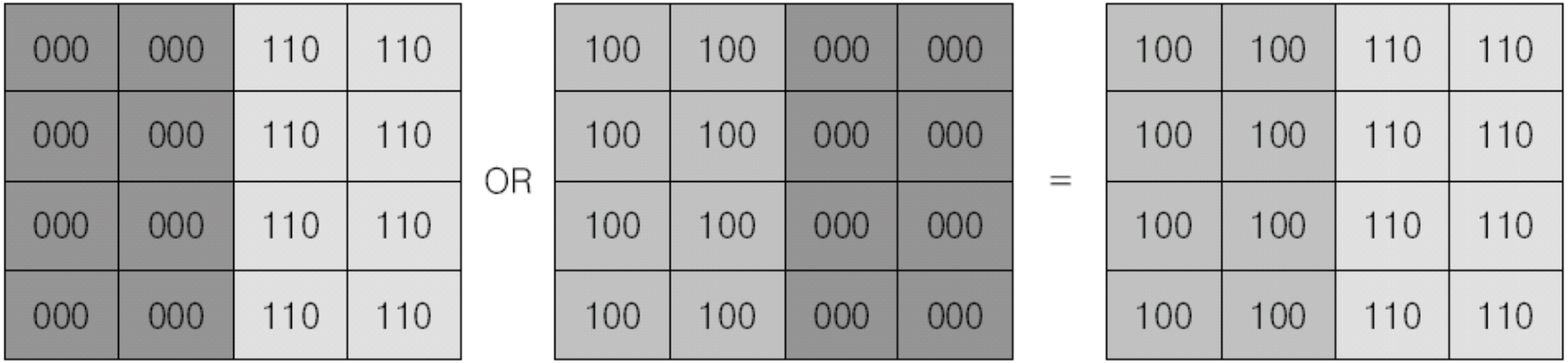


AND 연산으로 프레임 처리를 한 결과 영상

OR 연산을 이용한 프레임 처리

- 두 입력 데이터가 모두 0이면 0을 출력하고, 둘 중 하나라도 1이면 1을 출력하는 것
- 영상 하나에 다른 영상의 특정 부분을 추가할 때 사용됨.
- OR 연산을 이용한 프레임 처리 공식

  O(x, y) = I1(x, y) OR I2(x, y)



[그림 10-6] OR 연산을 이용한 프레임 처리의 특징

[실습하기 10-6] OR 연산을 이용한 프레임 처리 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_FRAME_OR
Caption	프레임 OR

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 프레임 단위 OR 연산을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnFrameOr
Doc Class	void	OnFrameOr

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 10-6] OR 연산을 이용한 프레임 처리 프로그램

```
void CImageProcessingDoc::OnFrameOr()
{
   CFile File;
   CFileDialog OpenDlg(TRUE);
   int i;
   unsigned char *temp;

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char[m_Re_size];

   if(OpenDlg.DoModal() == IDOK){
      File.Open(OpenDlg.GetFileName(), CFile::modeRead);
```

[실습하기 10-6] OR 연산을 이용한 프레임 처리 프로그램

```
        if(File.GetLength() == (unsigned)m_width * m_height){
            temp = new unsigned char[m_size];

            File.Read(temp, m_size);
            File.Close();

            // 프레임 간에 픽셀 대 픽셀로 OR 연산 실행
            for(i=0 ; i<m_size ; i++){
                m_OutputImage[i]
                    = (unsigned char)(m_InputImage[i] | temp[i]);
            }
        }
        else{
            AfxMessageBox("Image size not matched");
            return;
        }
    }
}
```

[실습하기 10-6] OR 연산을 이용한 프레임 처리 프로그램

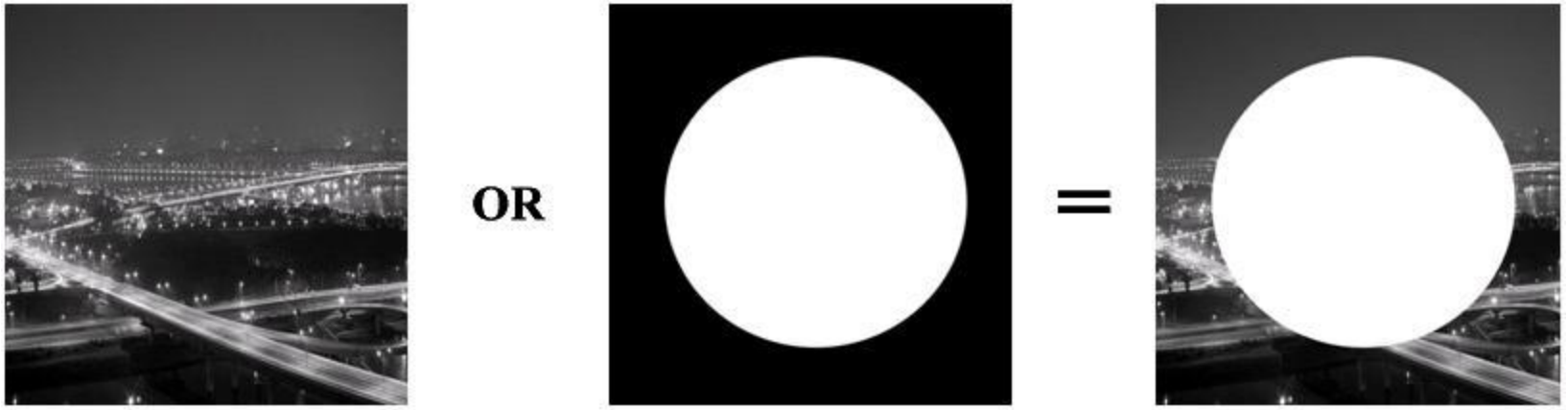
④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnFrameOr()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameOr();

    Invalidate(TRUE);
}
```

[실습하기 10-6] OR 연산을 이용한 프레임 처리 프로그램

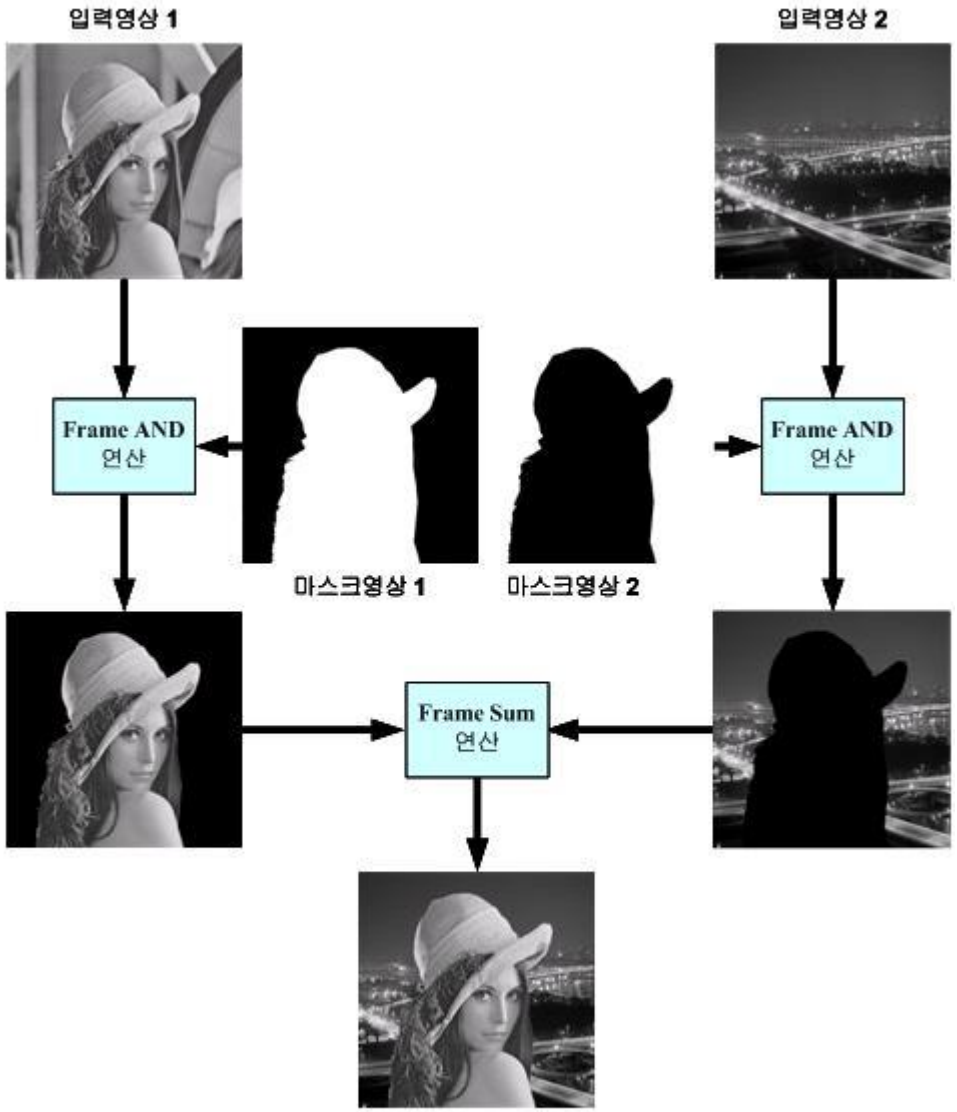
⑤ 프로그램 실행 결과 영상

- 실제 영상에 OR 연산을 이용한 프레임 처리를 수행하여 두 영상이 원하는 영역에서 서로 합성된 결과 영상



OR 연산으로 프레임 처리를 한 결과 영상

논리연산을 이용한 영상의 합성



[그림 10-7] 두 영상의 합성 과정

[실습하기 10-7] 논리연산을 이용한 영상 합성 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_FRAME_COMB
Caption	프레임 합성

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 프레임 합성을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnFrameComb
Doc Class	void	OnFrameComb

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 10-7] 논리연산을 이용한 영상 합성 프로그램

```
void CImageProcessingDoc::OnFrameComb()
{
    CFile File;
    CFileDialog OpenDlg(TRUE);
    int i;
    unsigned char *temp, *masktemp, maskvalue;

    m_Re_height = m_height;
    m_Re_width = m_width;
    m_Re_size = m_Re_height * m_Re_width;

    m_OutputImage = new unsigned char[m_Re_size];

    AfxMessageBox("합성할 영상을 입력하시오");

    if(OpenDlg.DoModal() == IDOK){ // 합성할 영상을 입력
        File.Open(OpenDlg.GetFileName(), CFile::modeRead);
        temp = new unsigned char[m_size];
        File.Read(temp, m_size);

        if((unsigned)m_width * m_height != File.GetLength()){
            AfxMessageBox("Image size not matched");
            // 영상의 크기가 같을 때
            return;
        }
        File.Close();
    }
    // 입력 영상, 합성할 영상, 마스크 영상의 크기가 같아야 한다.
```

[실습하기 10-7] 논리연산을 이용한 영상 합성 프로그램

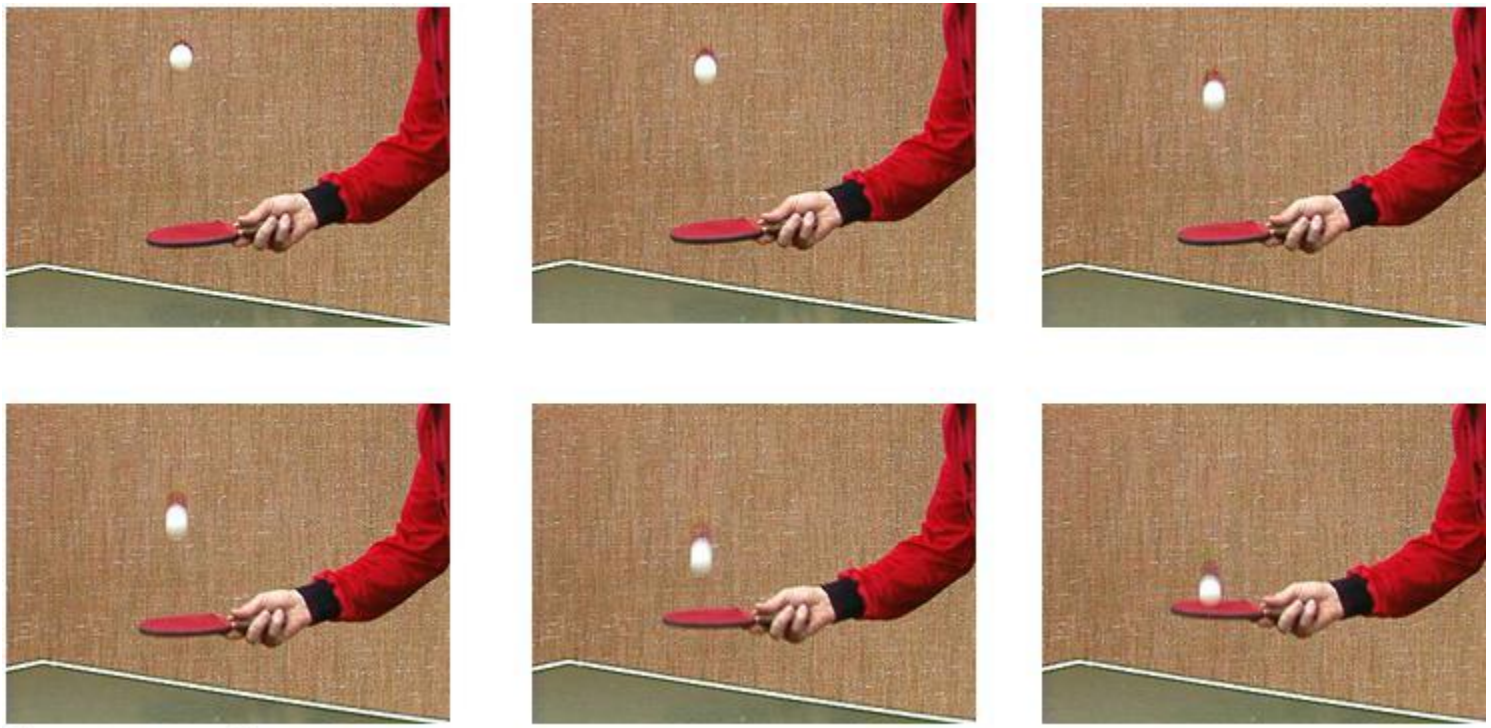
```
    AfxMessageBox("입력 영상의 마스크 영상을 입력하시오");
    if(OpenDlg.DoModal() == IDOK){ // 입력 영상의 마스크 영상
        File.Open(OpenDlg.GetFileName(), CFile::modeRead);
        masktemp = new unsigned char[m_size];
        File.Read(masktemp, m_size);
        File.Close();
    }

    for(i=0 ; i<m_size ; i++){
        maskvalue = 255 - masktemp[i];
        // 영상의 최대값에서 마스크 영상의 값을 뺀다.
        m_OutputImage[i]
            = (m_InputImage[i] & masktemp[i]) | (temp[i] & maskvalue);
        // 입력 영상과 마스크 영상은 AND 연산을 하고, 합성할 영상은
        // (255-마스크 영상) 값과 AND 연산을 실행한 후 두 값을 더한다.
    }
}
```

[실습하기 10-7] 논리연산을 이용한 영상 합성 프로그램

④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnFrameComb()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnFrameComb();

    Invalidate(TRUE);
}
```

- 동영상
  - 움직이는 순간적인 동작을 나타내는 정지영상을 순차적으로 보여주어 움직이는 것처럼 지각하게 한 것
  - 정지영상이 모여 동영상을 만듦
- 프레임
  - 동영상을 구성하는 개개의 정지영상



[그림 10-8] 동영상의 구성

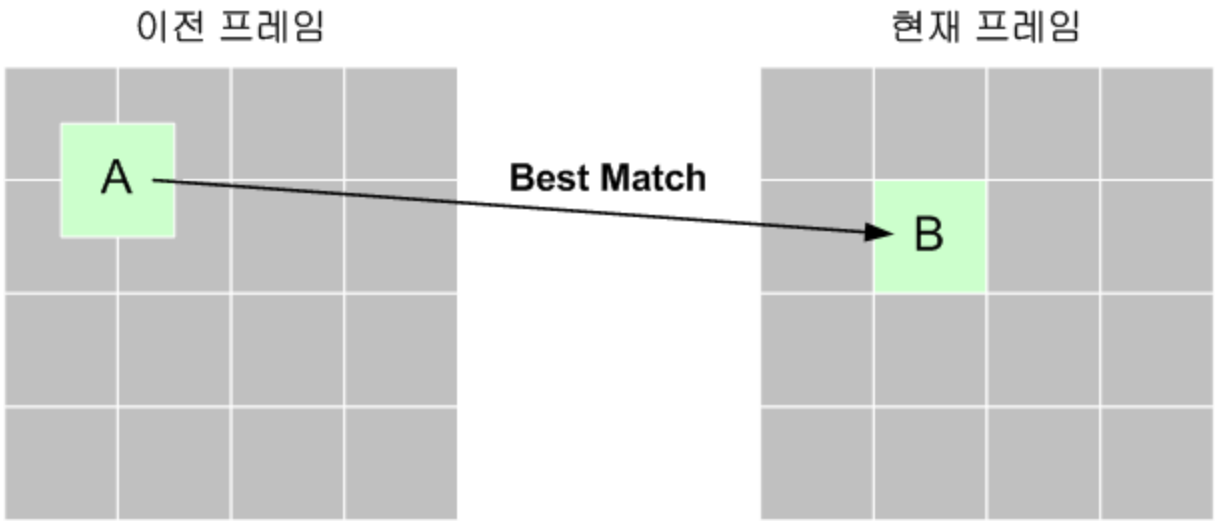
프레임 간 예측 (Frame to Frame Prediction) 기법

- 프레임 간 예측 (Frame to Frame Prediction) 기법
  - 연속적인 프레임 중에서 인접한 프레임은 서로 중복되는 부분이 많은데 이 중복 부분을 제거하여 데이터양을 줄이는 방법
  - 중복 데이터가 제거되어 데이터양이 현저하게 줄어듦.

- 움직임 예측(Motion Prediction)
  - 프레임을 몇 개의 블록으로 분할한 뒤 현재 프레임의 B 블록이 이전 프레임의 어느 곳에서 왔는지 찾는 과정을 수행
  - 즉, 가장 비슷하게 정합되는 A블록을 찾음.

- 움직임 보상(Motion Compensation)
  - 움직임 벡터에서 현재 프레임의 해당 블록을 생성하는 것
    - 찾은 A 블록에서 B 블록까지 가로축으로 이동한 거리와 세로축으로 이동한 거리를 구한 값을 움직임 벡터(Motion Vector)라고 하며, 이 값은 똑같은 내용이 어느 곳에 있는지를 알려줌.
    - 이 값만 저장하거나 전송하면 나중에 이전 프레임에서 현재 프레임의 B 블록에 A 블록을 복사하여 쓸 수 있음.

프레임 간 예측 (Frame to Frame Prediction) 기법(계속)

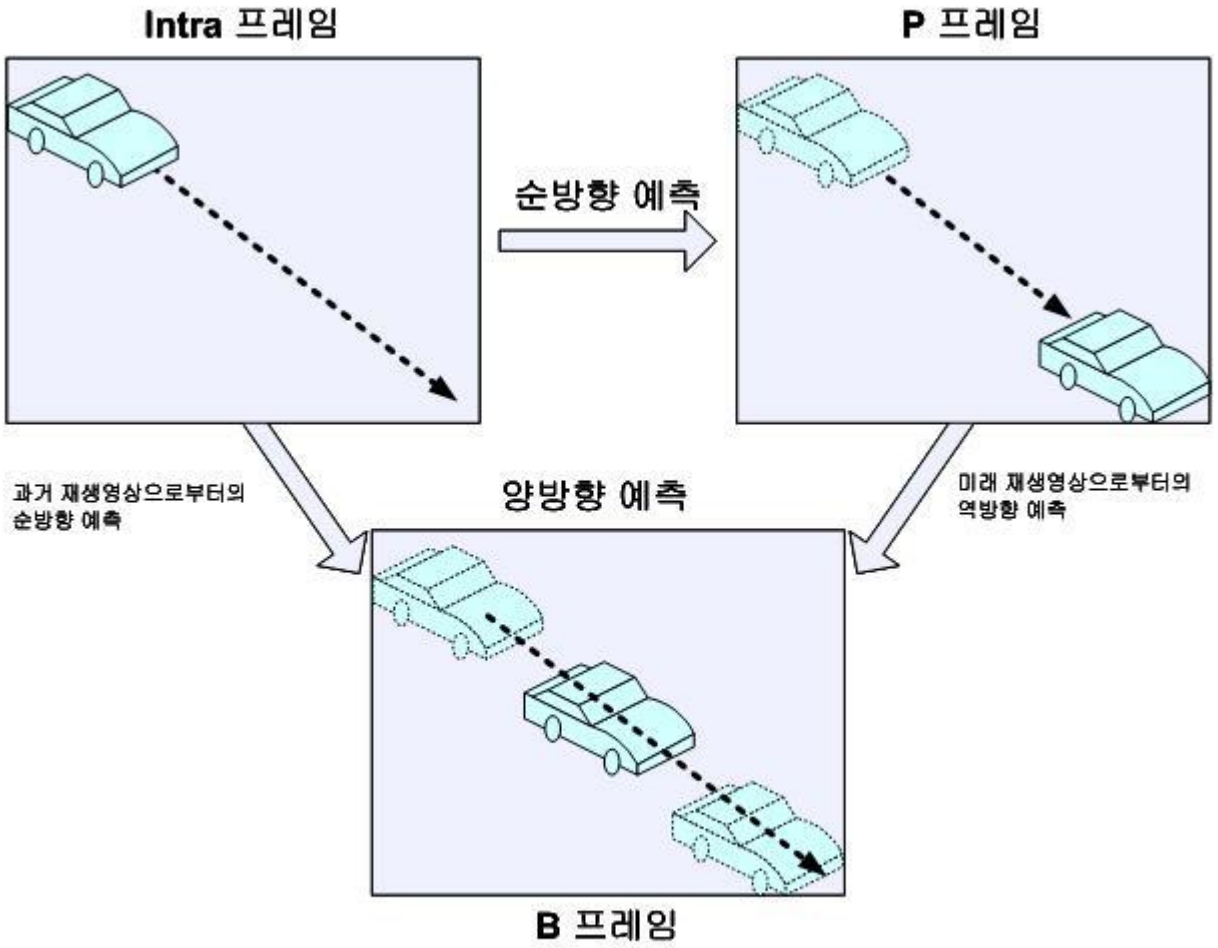


[그림 10-9] 프레임 예측의 수행 과정

프레임 간 예측에서 프레임의 종류

- 프레임처럼 예측 없이 모든 정보가 들어 있는 프레임과 예측으로 생성되는 프레임이 있음.
- I 프레임
  - I Picture(Intra-Picture)
  - 다른 프레임이 예측할 수 있도록 모든 정보를 그대로 저장하거나 전송해야 하는 프레임으로 독립적으로 복원이 가능
- P 프레임
  - P Picture(Predicted Pictures)
  - I 프레임 정보에서 예측 과정을 거쳐 프레임 생성
  - I 프레임에서 예측 영상을 생성하는 것은 순방향 예측 부호화, P 프레임은 순방향 예측 부호화 영상
- B 프레임
  - B Picture(Bi-directional predicted Pictures)
  - I 프레임과 먼저 생성된 P프레임에서 정보를 얻어 프레임 생성
  - 두 군데서 참조가 되므로 쌍방향 예측 부호화 영상이라고 함.

프레임 간 예측에서 프레임의 종류(계속)



[그림 10-10] 순방향과 쌍방향 예측

요약

- 프레임 처리
  - 두 개나 그 이상의 서로 다른 영상을 포함한 영상 간의 연산을 바탕으로 새로운 화소 값을 생성하는 것
- 프레임
  - 프레임 처리에서 입력으로 사용되는 각 영상
- 동영상
  - 움직임이 있는 디지털 영상을 연속적으로 보여 줘서 자연스런 움직임을 만들어 냄
  - 움직임이 있는 각 영상은 프레임
- 결합 처리
  - 서로 관련 있는 여러 영상을 합성하여 향상된 품질의 영상을 만드는 것으로, 주로 산술연산으로 수행
  - 영상 간의 덧셈, 뺄셈, 평균연산 등이 대표적
- 합성 처리
  - 새로운 영상을 만들려고 영상의 영역이나 연관성이 없는 영상을 혼합하는 것
  - 영상간의 **AND**나 **OR** 등이 대표적

요약

- 프레임간 예측 기법
  - 프레임이 연속해 있을 때 인접한 프레임에는 서로 같은 부분이 많은데, 이 중복 부분을 제거하여 데이터양을 줄이는 방법
- 움직임 예측
  - 현재 프레임의 블록이 이전 프레임의 어느 곳에서 왔는지를 찾는 과정
- 움직임 벡터
  - 찾은 이전 프레임의 블록에서 현재 프레임의 블록이 가로축으로 이동한 거리와 세로축으로 이동한 거리를 구한 값
- 움직임 보상
  - 움직임 벡터에서 현재 프레임의 해당 블록을 생성하는 것
- I 프레임
  - 다른 프레임이 예측할 수 있도록 모든 정보를 그대로 저장하거나 전송해야 하는 프레임으로 독립적으로 복원이 가능
- P 프레임
  - I 프레임 정보에서 예측 과정을 거쳐 프레임을 생성
  - I 프레임에서 예측 영상을 생성하는 것을 순방향 예측 부호화라 함
- B 프레임
  - I 프레임과 먼저 생성된 P 프레임에서 정보를 얻어 프레임 생성
  - 두 군데서 참조가 되므로 쌍방향 예측 부호화 영상이라고 함.

Thank you