http://dx.doi.org/10.5392/JKCA.2016.16.11.525

웹 사용자 인터페이스 향상을 위한 풀다운메뉴에서 방향제어가 가능한 애니메이션 기법

Animation Techniques with Direction Control in Pull-down Menu for Improving Web User Interface

조한수
한중대학교 공학부 모바일IT공학전공

Han-Soo Cho(hscho@hanzhong.ac.kr)

요약

최근 인터넷을 통한 정보획득이 증가함에 따라 웹을 기반으로 하는 사용자 인터페이스의 중요성이 그 어느 때 보다 강조되고 있다. 본 연구는 웹 사용자 인터페이스 향상을 위하여 풀다운메뉴에서 서브메뉴 슬라이딩 구조를 개선하는 것을 목적으로 한다. 많은 웹 사이트에서 사용 중인 풀다운메뉴의 서브메뉴에 적용된 애니메이션 기법은 매우 단조로운 슬라이딩 구조를 하고 있다. 이러한 문제점을 개선하기 위하여 풀다운메뉴에서 방향제어가 가능한 확대/축소 및 이동 애니메이션 기법에 기반을 둔 새로운 슬라이딩 알고리즘을 제안한다. 연구 결과, 제안한 방법은 기존의 풀다운메뉴와 비교하여 확연히 시각적인 효과를 향상할 수 있을 뿐만 아니라 웹을 기반으로 하는 사용자 인터페이스 구현에도 쉽게 적용할 수 있다. 제안한 기법을 반응형 이미지 슬라이더에 적용하는 실험에서도 좋은 결과를 얻었다. 향후 제안한 기법을 이용한 웹 기반 인터랙티브 콘텐츠를 구현하기 위하여 성능을 고려한 연구가 필요하다.

■ 중심어 : | 풀다운메뉴 | 슬라이딩 알고리즘 | 애니메이션 기법 | 웹 사용자 인터페이스 |

Abstract

As obtaining information via the Internet has increased recently, the importance of web-based user interface more than ever is being emphasized. The purpose of this study is to improve the sliding structures of submenu in pull-down menu for improving web user interface. Animation techniques applied to the submenu of the pull-down menus that are being used at many web sites have very monotonous sliding structures. In order to solve these problems, a new sliding algorithm based on enlarging/reducing and moving animation techniques with direction control in pull-down menu is proposed. As a result, the proposed method can not only improve visual effects significantly but be also easily applied to the implementation of the web-based user interface in comparison with the previous pull-down menus. Finally, experiments on application of the proposed sliding algorithm to responsive image slider show that the proposed method can achieve good results. Further studies taking into account performance are needed to implement web-based interactive contents using the proposed method.

■ keyword : | Pull-down Menu | Sliding Algorithm | Animation Technique | Web User Interface |

접수일자 : 2016년 08월 05일
수정일자 : 2016년 09월 05일
심사완료일 : 2016년 09월 20일
교신저자 : 조한수, e-mail : hscho@hanzhong.ac.kr

I. 서 론

웹을 이용하는 사용자의 요구가 보다 다양해짐에 따라 웹 사용자 인터페이스의 역할이 그 어느 때보다 중요한 이슈로 대두되고 있고, 그에 따른 사용자 인터페이스를 향상시키고자하는 연구[1][2]도 활발히 진행되어 많은 성과를 올리고 있다. 또한, 이와 관련하여 웹 사용자 인터페이스 요소의 대표적인 내비게이션 메뉴에 관한 연구도 함께 진행되어 정보탐색을 위한 메뉴디자인의 역할과 관련된 연구[3][4], 웹 내비게이션이 사이트에 미치는 영향에 관한 연구[5-7], 웹 사용자 인터페이스 디자인과 관련된 연구[8][9] 등의 다양한 연구가 보고되고 있다. 얼마 전까지만 해도 플래시 같은 저작도구를 사용한 내비게이션 메뉴가 주류를 이루었지만 별도의 플러그인을 설치해야 하는 불편 때문에 최근에는 자바스크립트와 같은 스크립트언어로 제작된 메뉴가 주로 사용되고 있다.

최근 내비게이션 메뉴 중에서 가장 많이 사용하고 있는 풀다운메뉴에 구현된 애니메이션 기법을 분석해 보면 대부분 매우 단조롭고 거의 유사한 형태를 하고 있다. 주로 메인메뉴를 선택하여 펼쳐지는 서브메뉴는 jQuery에서 제공하는 .slideDown()/.slideUp() 메서드를 이용하여 위에서 아래로, 아래에서 위로 펼침과 접힘 처리를 하고 있다. 이러한 메서드는 사용은 매우 편리하지만 인터랙티브 웹 콘텐츠 구현이라는 측면에서 볼 때 새로운 방법이 절실히 요구된다. 또한 지금까지의 내비게이션에 관한 연구에서는 디자인측면이 강조된 내용이 대부분이고 웹 기반 UI 프로그래밍 기법에 근거를 둔 구현방법에 대한 연구는 거의 보고된 바 없다. 이에 본 논문에서는 웹 사용자 인터페이스 향상을 위하여 프로그래밍기법에 근거한 커스터마이즈한 애니메이션 기법을 웹 기반 콘텐츠 구현에 적용할 수 있도록 풀다운메뉴에서 방향제어가 가능한 애니메이션 기법을 제안한다.

제안한 기법은 확대/축소 및 이동 애니메이션 기법에 기반을 두고 있기 때문에 기존의 풀다운메뉴의 서브메뉴 펼침과 접힘 처리에 구현되어 있는 상, 하 방향의 애니메이션 기법을 사용한 단조로운 슬라이딩 구조와 비교하여 확연히 차별화되며, 또한 시각적인 효과를 향상할 수 있어서 웹 사용자 인터페이스 구현을 위한 새로운 기술적 요소를 제공한다. 또한, 본 논문에서 제안한 기법은 풀다운메뉴의 서브메뉴처리에서 뿐만 아니라 웹을 기반으로 하는 콘텐츠 제작에 적용되어 콘텐츠 고유의 기능적인 측면을 향상할 수 있다. 이러한 효율성과 확장성을 검증하기 위하여 제안한 애니메이션 기법을 이용한 슬라이딩 알고리즘을 풀다운메뉴와 반응형 이미지 슬라이더 구현에 적용하여 그 유효성을 확인하였다.

II. 이론적 배경

1. CSS(Cascading Style Sheets)

HTML 엘리먼트만을 사용하여 웹 페이지를 제작하는 경우에는 많은 노력과 제약이 따른다. 또한 유지 보수도 매우 어렵다. 이러한 이유로 최근 웹 페이지 제작을 위해 W3C의 표준인 CSS[10]가 많이 이용된다. CSS는 HTML 엘리먼트에 스타일을 적용하여 다양한 레이아웃을 구현할 수 있는 웹 기술로서, HTML 문서가 랜더링되어 출력되는 방법을 기술한 언어이다. CSS는 HTML 엘리먼트를 선택하는 선택자와 선택된 엘리먼트에 적용할 속성을 선언한 선언블록으로 구성된 CSS 규칙집합을 웹 문서에 포함시켜 사용한다. 문서가 로딩되면 선언된 스타일이 HTML문서에 적용된다. 규칙의 속성이 간결하고, 시각적 기능이 강력하여 최근 웹을 기반으로 하는 콘텐츠 제작에 중요한 부분을 차지하고 있는 기술이다.

2. 엘리먼트 배치를 위한 CSS의 position 속성

CSS에서 position 속성(relative, absolute, static, fixed)은 HTML로 작성된 웹문서에서 엘리먼트를 배치하는 방식을 지정하는 속성이다. relative 방식은 현재 엘리먼트가 정상적인 흐름에 의해 본래 놓여있었던 자신의 위치를 기준으로 오프셋만큼 이동되어 배치되며 다른 엘리먼트에 영향을 주지 않는 위치 지정 방식이다. 이에 비해 absolute 방식은 position(static 제외)이

지정된 첫 번째 조상 엘리먼트를 기준으로 오프셋만큼 떨어져 배치되는 방식으로 문서의 일반적인 흐름에 의해 배치되지 않기 때문에 다른 엘리먼트에 영향을 준다. 본 논문에서는 제안한 기법의 애니메이션 처리를 위해 relative 방식과 absolute 방식의 상호관계를 이용하여 엘리먼트를 배치한다.

3. jQuery를 이용한 이벤트처리

jQuery[11]는 자바스크립트 라이브러리로서 최근 CSS와 혼용하여 인터랙티브 웹 콘텐츠 제작을 위해 많이 사용된다. 또한 문법이 매우 간결하기 때문에 자바스크립트로 코딩하는 것에 비해 많은 노력을 경감할 수 있는 장점이 있다. 또한, 메뉴에 마우스가 올라가거나 내려갈 때 발생하는 mouseenter/mouseleave 이벤트에 대한 처리방법도 비교적 간단하여 본 논문에서 제안한 방법을 기술하는 프로그래밍 요소로서 CSS와 jQuery를 이용한다. CSS에서는 애니메이션을 진행하기 위해 엘리먼트의 상태를 설정하고, 애니메이션 구현을 위한 이벤트처리는 jQuery를 이용한다.

Ⅲ. 방향제어가 가능한 애니메이션 기법

프로그래밍기법을 이용하여 풀다운메뉴의 구조를 개선한 연구[12]에서는 하이라이트이미지 애니메이션 기법을 사용하여 시각적 효과를 향상하는 방법을 제안하였다. 이 방법은 메인메뉴를 구성하는 각 엘리먼트에 대하여 쌓임을 구성해 놓고 하이라이트이미지 애니메이션 기법을 이용하여 선택된 메인메뉴로 슬라이딩을 진행함으로써 시각적 효과를 향상하는 방법으로, 메인메뉴를 선택할 때 시각적인 효과 향상을 보이고 있다, 그러나 이 기법은 사용자 정보선택이 실제로 이루어지는 서브메뉴처리에서는 단순한 동작을 하고 있다. 즉, 메인메뉴가 선택되었을 때 해당 서브메뉴는 위에서 아래로 펼쳐지고, 선택이 해제되었을 때는 펼쳐진 서브메뉴가 아래에서 위로 접히는 단순한 한 가지 동작으로만 서브메뉴를 처리하고 있다. 이와 같은 서브메뉴 처리의 단순한 동작은 현재 웹 사이트에서 서비스되고 있는 풀다운메뉴에서도 동일한 형태를 띠고 있다. 이에 본 논문에서는 기존연구방법에서 사용한 일률적인 서브메뉴 펼침과 접힘 처리방식의 문제점에 대하여 웹 사용자 인터페이스 향상을 위한 방향제어가 가능한 사용자 중심의 새로운 커스터마이즈한 서브메뉴 애니메이션 기법을 제안한다. 제안한 기법은 풀다운메뉴에서 메인메뉴 및 서브메뉴의 애니메이션 처리뿐만 아니라 사용자 중심의 다양한 인터랙티브 웹 콘텐츠 제작에 필요한 펼침, 접힘, 슬라이딩과 같은 애니메이션 동작을 구현할 때에도 쉽게 적용 가능하다.

1. HTML 구성

[그림 1]은 본 논문에서 제안한 기법을 풀다운메뉴의 서브메뉴처리에 적용하기 위해 1단계서브메뉴와 2단계서브메뉴로 구성된 풀다운메뉴의 메인메뉴리스트 1개를 표현하는 HTML이다. 1단계서브메뉴는 'ul-child-submenu' 클래스를 가지는 ul 엘리먼트(이하 .ul-child-submenu 엘리먼트)이며, 그 안에 6개의 서브메뉴리스트아이템(C1. HTML5~C6. Web Design)을 포함하고 있다. 2단계 및 그 이하 서브메뉴에는 'ul-desc-submenu' 클래스를 설정하여 1단계서브메뉴는 메인메뉴에서 아래로 펼쳐지고, 2단계이상 서브메뉴는 우측에서 펼쳐지는 구조적 특징을 고려하였다. 또한 서브메뉴에 공통적으로 속성을 지정하는 경우를 고려하여 'ul-submen' 클래스를 이중으로 설정하였다.

```
<ul class="ul-topmenu">
    .....
  <li class="li-has-child-submenu">
    <a>Categories</a>
    <ul  class="ul-child-submenu ul-submenu">
      <li><a>C1.HTML5</a></li>
      <li><a>C2.CSS</a></li>
      <li class="li-has-desc-submenu">
        <a>C3.jQuery Basic</a>
        <ul class="ul-desc-submenu ul-submenu">
          <li><a>C3.1 Selector</a></li>
          <li><a>C3.2 Traversing</a></li>
          <li><a>C3.3 Manipulation</a></li>
          <li><a>C3.4 jQuery Effects</a></li>
          <li><a>C3.5 jQuery Events</a></li>
          <li><a>C3.6 CSS Styling</a></li>
        </ul>
      </li>
      <li><a>C4.Javascript</a></li>
      <li><a>C5.Web Development</a></li>
      <li><a>C6.Web Design</a></li>
    </ul>
  </li>
    .....
</ul>
```

그림 1. 풀다운메뉴의 서브메뉴를 구성하는 HTML

2. 제안한 애니메이션 기법의 이론적 원리

이 절에서는 기존의 서브메뉴 애니메이션 처리의 문제점을 분석하고 제안한 방향제어가 가능한 애니메이션 기법에 필요한 논리를 도출하여, 이를 바탕으로 기존의 방법의 문제점을 해결하는 제안한 방법의 이론적 원리에 대해 기술한다.

2.1 기존방법에서의 서브메뉴 애니메이션 문제점

풀다운메뉴의 서브메뉴 펼침과 접힘 처리에 다양한 방향의 애니메이션을 적용하기 위해서는 먼저, 서브메뉴가 보이지 않는 상태에서 메인메뉴에 마우스가 올라갔을 때 서브메뉴가 점점 애니메이션으로 확대되면서 출력되어야 한다. 이러한 전제조건을 만족하기 위해서 문서가 로딩되기 전 미리 서브메뉴를 보이지 않는 상태로 설정해두고 메인메뉴에 마우스가 올라가면 서브메뉴를 보이는 상태로 변경하고 확대 애니메이션을 진행하면 간단히 해결될 것 같지만 서브메뉴가 보이는 상태로 변경되면 확대 애니메이션은 더 이상 진행할 필요가 없다(이미 보이는 상태로 변경되었기 때문에). 또는 보이지 않는 상태에서 확대 애니메이션을 진행하는 것은 의미가 없어서 서브메뉴 처리 전제조건을 만족하지 못한다. 서브메뉴 펼침과 접힘 처리에 애니메이션을 적용하는 문제는 그 전제조건이 메인메뉴가 선택되기 전 서브메뉴는 항상 보이지 않는 상태이어야 된다는 점이 일반적인 확대애니메이션과 확연히 다른 점이며, 또한 사용자 요구에 따른 다양한 방향의 애니메이션을 서브메뉴처리에 적용하는 것을 어렵게 한다. 실제 웹 사이트에서 구현된 풀다운메뉴에서의 서브메뉴 펼침과 접힘 처리에서도 상하방향의 단조로운 애니메이션 처리만 가능한 jQuery에서 제공하는 메서드(.slideDown()/.slideUp())에만 의존하고 있다 이러한 메서드를 사용하는 방법으로는 서브메뉴 애니메이션처리가 상하방향으로 고정되어 있기 때문에 다양한 방향으로 서브메뉴 펼침과 접힘 애니메이션을 제어할 수 없을 뿐만 아니라 다른 웹 기반 인터랙티브 콘텐츠 구현에 적용하는 것도 아주 제한적일 수밖에 없다. 이러한 문제점을 해결하고자 하는 연구도 거의 보고되지 않고 있다. 최근에 보고된 애니메이션 기법을 사용하여 풀다운메뉴의 구조를 개선한 연구[12]에서도 기술한 문제점과 동일한 상하방향의 슬라이딩으로 서브메뉴의 펼침과 접힘 처리를 하고 있다. 이러한 이유로 다양한 방향으로 서브메뉴의 펼침과 접힘 애니메이션 처리를 구현하는 것은 어려운 문제점으로 분류된다.

2.2 제안한 방법의 서브메뉴 확대 애니메이션

본 논문에서는 이러한 문제점을 해결하기 위한 수단으로 먼저, 다음과 같은 HTML에서 엘리먼트의 관계에 대한 이론적 논리를 도출하고 이를 응용하여 서브메뉴 애니메이션 처리의 문제점을 해결한다.

i) [그림 1]의 HTML구성에서처럼 서브메뉴를 표현하는 ul(.ul-child-submenu) 엘리먼트는 서브메뉴리스트를 표현하는 li 엘리먼트 아래의 레이어에 위치한다. 이것은 DOM 구조에서 부모엘리먼트보다 자식엘리먼트가 상위 레이어에 놓여서 쌓임을 형성하기 때문이다(상위 레이어에서 하위 레이어 방향으로 브라우저에 보이게 된다).

ii) 하위 레이어에 위치하는 부모엘리먼트인 ul 엘리먼트 영역크기(width, height)에 상관없이('0'이 되어도) visibility 속성이 'visible'이면 자식엘리먼트의 전체영역은 브라우저에 출력되어 보이게 된다.

iii) ul 엘리먼트의 visibility 속성이 'hidden'이면 DOM상에서 영역은 차지하지만 보이지 않는 상태이다. 이 상태에서의 애니메이션은 서브메뉴 애니메이션 처리의 전제조건에 부합되지 않는다.

iv) 엘리먼트가 'hidden'상태에서 'visible'상태로 변경되면, 변경이 완료되는 순간에 ii)에서 기술한 것과 같이 ul 엘리먼트 영역크기에 상관없이 li 엘리먼트(서브메뉴리스트아이템)는 단번에(애니메이션 없이) 전체영역이 출력되어 브라우저에 나타나게 된다.

본 논문에서는 위에서 도출된 부모-자식 엘리먼트의 쌓임과 visibility 속성과의 관계를 이용하여 서브메뉴리스트아이템을 감싸고 있는 ul 엘리먼트를 제어함으로써 서브메뉴 애니메이션 전제조건을 만족하는 확대

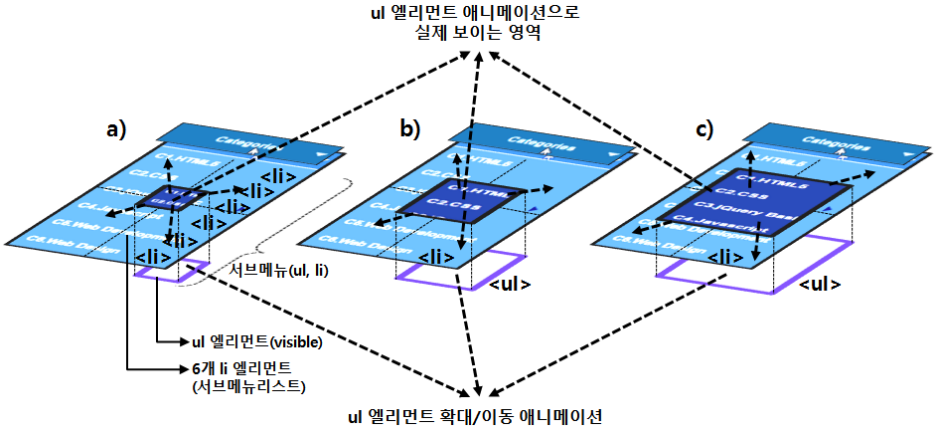

그림 2. 확대 및 이동애니메이션 과정

애니메이션 처리가 가능하다. 이를 위해 먼저, 서브메뉴 리스트가 초기에는 보이지 않는 상태에서 애니메이션을 진행해야 하므로, ul 엘리먼트의 visibility 속성을 'hidden'으로 설정하고, 영역크기(width, height)도 '0'으로 초기화한다. 이 상태에서는 서브메뉴리스트는 보이지 않는다. 이러한 조건에서 메인메뉴에 마우스가 올라가서 ul 엘리먼트의 visibility 속성을 'hidden'에서 'visible'로 변경하는 것은 ii)와 iv)에서 기술한 것처럼 ul 엘리먼트 영역의 크기와 상관없이 서브메뉴리스트는 애니메이션 없이 단번에 나타나게 된다. 그 이후에 확대애니메이션을 진행하는 것은 이미 서브메뉴가 출력되었기 때문에 아무런 의미를 갖지 못한다. 따라서 이러한 문제에 대하여 제안한 방법에서는 'hidden'상태인 ul 엘리먼트를 'visible'로 변경하는 처리와 영역크기에 대한 확대애니메이션처리를 동시에 진행한다. 이러한 동시처리방법은 두 조건이 별개로 처리되지 않기 때문에 확대애니메이션이 진행 중에는 ul 엘리먼트의 visibility 속성이 visible 상태가 완료된 상태가 아니므로 iv)조건에 부합되지 않고, 또한 'hidden'상태에서 확대애니메이션을 진행하는 상태도 아니므로 iii)조건에도 해당되지 않는 결과가 된다. 즉, 이러한 진행은 지정한 애니메이션 속도에 따라 ul 엘리먼트의 width와 height 속성 값이 각각 '0'에서 시작하여 우측과 아랫방향으로 점점 영역 확대가 진행되는 형태에 따라 자식 엘리먼트인 서브메뉴리스트가 보이게 되는 슬라이딩 처리가 이루어진다.

2.3 제안한 방법의 서브메뉴 이동 애니메이션

확대애니메이션에 이동애니메이션을 추가하여 이를 동시에 진행하면 애니메이션 방향을 제어할 수 있다. 즉, 확대애니메이션은 엘리먼트가 놓여 있는 좌표가 기준이 되어 우측 또는 아래로 확대가 진행된다. 서브메뉴의 애니메이션 방향을 제어하기 위해서는 확대가 진행되는 순간에 이동애니메이션을 함께 진행해야 한다. 이는 초기 엘리먼트가 놓이는 시작위치와 애니메이션을 종료할 종료위치를 설정하면 확대애니메이션과 이동애니메이션은 동시에 진행되어 슬라이딩이 이루어진다. 이것은 확대 애니메이션에서 visibility 속성 변경과 애니메이션을 동시에 진행하는 것과 동일한 이치이다. [그림 2]에서는 상위 레이어에 li 엘리먼트(서브메뉴리스트아이템)가 놓이고, 그 아래 레이어에 ul 엘리먼트가 배치되어, 서브메뉴 정중앙에 위치한 ul 엘리먼트가 방사형 형태로 확대 및 이동 애니메이션이 진행됨에 따라 자식엘리먼트인 서브메뉴리스트가 애니메이션 되는 형상 그대로 출력되는 과정을 보이고 있다. 초기 ul 엘리먼트의 위치(좌측상단모서리)는 서브메뉴의 정중앙 좌표 점이고 애니메이션 종료위치는 메인메뉴 좌측아래 좌표 점이다. 만약 종료위치를 설정하지 않으면 서브메뉴의 정중앙 좌표 점을 기준으로 우측 아랫방향으로만 확대애니메이션만 진행되고 이동애니메이션을 일어나

지 않는다. 본 논문에서는 이러한 처리과정으로 기존의 방법[12]에서 사용하는 상하방향의 슬라이딩방법을 개선할 수 있다. 정리하면, 엘리먼트의 visibility 속성 및 확대애니메이션에 필요한 width와 height 속성과, 그리고 이동애니메이션을 진행할 시작위치와 종료위치에 대한 속성에 대하여 애니메이션을 동시에 진행하면 ul 엘리먼트의 영역이 확대됨으로써 그 안에 포함되어 있는 서브메뉴리스트의 영역도 확대영역에 따라 나타나게 되고, 그 순간에 거의 동시에 정해진 위치로 이동애니메이션이 함께 이루어지므로 서브메뉴는 다양한 방향으로 슬라이딩이 가능해진다. 본 논문에서 제안한 방법을 이용하면 풀다운메뉴에서 뿐만 아니라 웹을 기반으로 하는 콘텐츠 구현에 쉽게 적용할 수 있다.

3. 확대/축소 애니메이션을 이용한 서브메뉴 슬라이딩 알고리즘

[그림 3]은 본 논문에서 제안한 애니메이션 기법에 의하여 서브메뉴가 상하좌우 방향, 대각선 방향, 양방향으로 슬라이딩되는 12가지 애니메이션 유형을 나타낸다. 화살표방향은 서브메뉴 펼침 처리이고, 그 역방향은 접힘 처리이다(제안한 12가지 방향 중에서 기존의 방법에서는 a-1 방향으로만 슬라이딩 처리를 하고 있다). 이와 같은 애니메이션 방향을 제어하기 위해서는 애니메이션을 진행할 시작위치와 종료위치, 확대/축소 및 이동애니메이션을 위한 속성들의 설정, 그리고 서브메뉴를 슬라이딩하는 알고리즘이 필요하다. 이 12가지 애니메이션 유형에서는 확대/축소 애니메이션을 적용해야 되는 유형과 확대/축소 및 이동애니메이션을 동시에 적용해야 하는 유형으로 분류된다. 전자는 a-1, a-3, b-1 유형으로, 서브메뉴 사각형영역의 좌측상단 모서리가 메인메뉴리스트 하단 좌측에 위치하는 경우이다. 나머지 유형은 확대/축소 애니메이션과 이동애니메이션을 동시에 진행해야 한다.

먼저, a-1, a-3, b-1 유형의 1단계서브메뉴에 대한 확대/축소 애니메이션을 이용한 슬라이딩 알고리즘은 다음과 같다. step1부터 step3까지의 설정은 CSS에 규칙으로 삽입되어, 문서 로딩과 함께 해당 엘리먼트에 적용된다. step4는 문서로딩이 완료되어 메인메뉴리스트

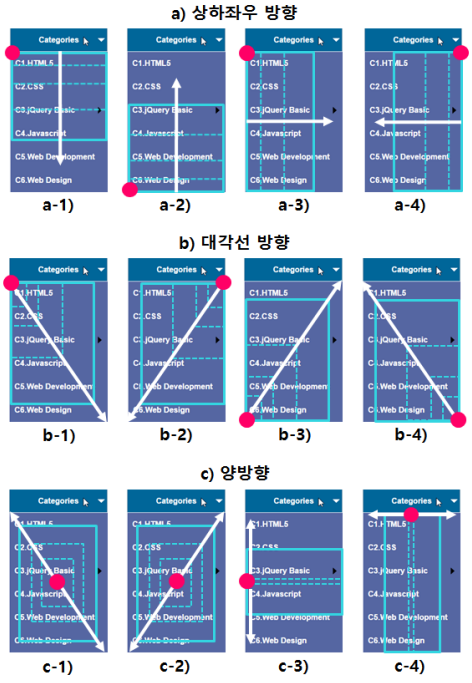

그림 3. 서브메뉴 애니메이션 방향

에 mouseenter 이벤트가 발생했을 때 슬라이딩처리이고, step5는 mouseleave 이벤트에 대한 처리이다.

[step1] 1단계서브메뉴의 초기 위치를 설정한다.

(step1-1) 메인메뉴리스트 .li-has-child-submenu 엘리먼트와 그 안에 포함된 1단계서브메뉴인 .ul-child-submenu 엘리먼트의 position 속성을 각각 relative와 absolute로 지정함으로써, 1단계서브메뉴는 메인메뉴리스트를 기준으로 배치된다.

(step1-2) 1단계서브메뉴의 오프셋을 지정한다. 오프셋은 메인메뉴리스트아이템("Categories")의 좌측 상단 모서리 좌표를 기준으로 설정되므로 left 속성은 '0'이 되고, top 속성은 '메인메뉴리스트높이'가 된다. 이와 같은 position 관계를 설정함으로써 부모엘리먼트인 li 엘리먼트(메인메뉴 또는 서브메뉴 리스트아이템)를 기준으로 서브메뉴 위치를 설정할 수 있기 때문에 서브메뉴가 깊어지더라도 항상 일관된 위치에 배치할 수 있다.

[step2] 1단계서브메뉴 .ul-child-submenu 엘리먼트의 visibility 속성을 'hidden'으로 설정하여 그 안에 포함된 서브메뉴리스트아이템([그림 1]에서 'C1.HTML

5~C6.Web Design')을 표현하는 6개의 li 엘리먼트를 보이지 않게 설정한다. 보이지 않지만 문서에서 엘리먼트가 위치하는 공간은 차지하고 있는 상태이다.

[step3] 1단계서브메뉴인 .ul-child-submenu 엘리먼트에 애니메이션 방향에 따라 height와 width 속성을 지정한다. 즉, 위에서 아래로 확대 애니메이션 되는 a-1 유형은 height 속성을 '0'으로, 왼쪽에서 오른쪽으로 슬라이딩되는 a-3 유형은 width 속성을 '0'으로, b-1 유형의 경우는 애니메이션 방향이 대각선이 되도록 height와 width 속성 모두를 '0'으로 설정한다. 지금까지 진행된 절차를 CSS로 기술하면 아래와 같다.

```
.li-has-child-submenu {
    position: relative;                  /* step1 */
}
.ul-child-submenu {
    position: absolute;                  /* step1 */
    left: 0; top: 메인메뉴리스트높이; /* step1 */
    visibility: hidden;                  /* step2 */
    height: 0;                 /* step3, a-1 경우 */
    width: 0;                  /* step3, a-3 경우 */
    height: 0; width: 0;    /* step3, b-1 경우 */
}
```

[step4] 이 단계는 메인메뉴에 mouseenter 이벤트가 발생할 때의 처리로서, 1단계서브메뉴에 대하여 확대 애니메이션에 의한 슬라이딩 처리를 진행한다.

(step4-1) 메인메뉴리스트 아이템인 .li-has-child-submenu 엘리먼트에 mouseenter 이벤트가 발생하면 먼저, 'hidden'으로 설정된 1단계서브메뉴 .ul-child-submenu 엘리먼트의 visibility 속성을 'visible'로 변경한다. 이것은 확대 애니메이션으로 슬라이딩이 진행되고 있는 상태가 보이도록 하기 위함이다.

(step4-2) .animate() 메서드를 사용하여 a-1유형은 height 속성을 '1단계서브메뉴높이'로, a-3 유형은 width 속성을 '100%'로, b-1 유형은 height 속성과 width 속성을 각각 '1단계서브메뉴높이'와 '100%'로 확대 애니메이션을 진행한다. 다음은 b-1유형을 예를 들어 step4-1과 step4-2에서 진행되는 애니메이션 처리에 대한 부분코드이다. 여기서 $subMenu는 jQuery에서 1단계서브메뉴 엘리먼트(.ul-child-submenu)를 읽어온 jQuery 객체이며 .css() 메서드는 스타일 속성을 변경하기 위한 jQuery 메서드이고, '%'단위는 '컨테이닝 블록의 width 속성 값에 대한 비율'을 의미한다.

```
$subMenu
    .css({visibility: "visible"})
    .stop(true, false)
    .animate({
        height: subMenuHeight,
        width: "100%"
    }, "fast");
```

step4-1과 step4-2의 처리는 동시에 진행되어서 확대 애니메이션에 의한 슬라이딩이 완성된다. 이미 기술한 바와 같이 부모엘리먼트인 .ul-child-submenu 엘리먼트는 영역의 크기에 상관없이 visibility 속성이 'visible'로 변경이 완료되면 서브메뉴리스트아이템은 단번에 보이게 되어 확대 애니메이션을 진행할 수 없다. 따라서 이 단계에서 .ul-child-submenu 엘리먼트의 visibility 속성을 'visible'로 변경함과 동시에 .animate() 메서드를 사용하여 애니메이션 유형에 따라 height/width 속성을 이용하여 애니메이션을 진행하면 지정한 속도에 따라 진행방향으로 확대 애니메이션이 되어 1단계서브메뉴는 진행되는 애니메이션 슬라이딩 형태와 동일하게 보이게 된다.

[step5] 메인메뉴에 mouseleave 이벤트가 발생하면 확대애니메이션이 진행되었던 1단계서브메뉴의 속성(height 및 width)에 대해 축소애니메이션으로 슬라이딩을 진행하여 이전의 상태로 환원한다. 애니메이션 종료 후 visibility 속성도 'hidden'으로 초기화함으로써 확대/축소 애니메이션에 의한 슬라이딩 처리는 완료된다.

4. 확대/축소 애니메이션과 이동 애니메이션을 이용한 확장된 슬라이딩 알고리즘

이동애니메이션은 확대애니메이션 방향과 반대방향

으로 슬라이딩이 필요한 경우이다(a-1, a-3, b-1유형을 제외한 유형). 이동애니메이션을 수행하기 위해서는 .animate() 메서드에서 left 속성과 top 속성으로 애니메이션 종료위치를 명시해야 한다. 다음은 앞에서 기술한 알고리즘에 이동애니메이션을 포함한 확장된 알고리즘이다.

[step1] 1단계서브메뉴의 초기 위치 즉 애니메이션이 진행되는 시작위치를 설정한다.

(step1-1) 확대/축소 알고리즘과 동일하다.

(step1-2) 확대/축소 및 이동애니메이션을 적용해야 하는 유형에서는 메인메뉴리스트의 좌측상단 모서리를 기준으로 1단계서브메뉴 사각형영역의 좌측상단 모서리가 놓이는 시작위치([그림 3]에서 작은 원)까지의 x, y축 방향의 좌표 값에 해당하는 거리만큼 left 속성과 top 속성을 지정한다. 특히, top 속성은 각 메인메뉴리스트에 포함된 1단계서브메뉴 높이가 일정하지 않은 경우를 고려하여 각 메인메뉴리스트에 mouseenter 이벤트가 발생했을 때 동적으로 1단계서브메뉴 높이를 계산하여 설정한다. 따라서 이 단계에서는 left 속성만 설정하고 top 속성은 step4에서 설정한다. left 속성은 a-2, b-3, c-3 유형은 '0'으로, c-1, c-2, c-4 유형은 '50%'로, a-4, b-2, b-4 유형은 '100%'로 설정한다.

[step2] 확대/축소 알고리즘과 동일하다.

[step3] 슬라이딩될 방향(수직, 수평, 대각선)에 따라 1단계서브메뉴의 height 속성과 width 속성을 CSS에서 '0'로 설정한다.

1) a-2, c-3 : .ul-child-submenu {height: 0;}

2) a-4, c-4 : .ul-child-submenu {width: 0;}

3) b-2, b-3, b-4, c-1, c-2 :
.ul-child-submenu {height: 0; width: 0;}

[step4] 지금까지 설정한 CSS가 포함된 HTML 문서가 로딩되고, 메인메뉴리스트에 mouseenter 이벤트가 발생하면 다음단계를 진행한다.

<step4-1> 1단계서브메뉴의 시작위치 중 top 속성을 구한다(left 속성은 이미 CSS에 설정되어 있음). top 속성의 위치(startTopPosition)는 아래와 같이 메인메뉴리스트높이(mainMenuHeight)에, 1단계서브메뉴높이(subMenuHeight)와 topFactor를 곱한 값을 더함으로써 구할 수 있다.

startTopPosition = mainMenuHeight +
subMenuHeight × topFactor

여기서 topFactor는 애니메이션 유형에 따른 시작위치 조절변수이다. 즉, a-4, b-2, c-4 유형의 topFactor 값은 '0'으로 설정하여 startTopPosition은 메인메뉴높이가 되고, a-2, b-3, b-4 유형은 topFactor 값이 '1'이 되어 startTopPosition은 메인메뉴리스트높이에 1단계서브메뉴높이를 더한 위치가 된다. c-1, 2, 3 유형의 topFactor는 '0.5'가 된다.

<step4-2> top 속성의 값이 구해지면 아래와 같이 top 속성에 따른 1단계서브메뉴의 위치를 변경한다.

$subMenu.css({top: startTopPosition})

<step4-3> 1단계서브메뉴 .ul-child-submenu' 엘리먼트의 visibility 속성을 'hidden'에서 'visible'로 변경한다(확대/축소 알고리즘과 동일하게 아래조건과 동시에 진행된다).

<step4-4> 이 단계에서는 이전 단계에서 설정된 조건을 기초로 하여 확대/축소 및 이동 애니메이션을 동시에 진행한다. 이를 위한 처리절차는 다음과 같다. 각 단계는 순차적으로 진행되는 것이 아니라 각 조건을 통합하여 확대/축소 및 이동 애니메이션이 동시에 이루어진다.

(step4-4-1) 슬라이딩 형태가 수직인 유형(a-2, c-3)은 height 속성을 'subMenuHeight' 만큼, 수평인 유형(a-4, c-4)은 width 속성을 '100%' 만큼 확대 애니메이션을 진행한다. 그리고 대각선 방향으로 슬라이딩되는 b-2, b-3, b-4 유형과 c-1, c-2 유형에 대해서는 height 속성과 width 속성을 각각 'subMenuHeight'와 '100%'로 설정하여 확대 애니메이션을 진행한다.

(step4-4-2) 1단계서브메뉴에 설정한 시작위치 중에서 left 속성이 '0'인 경우에는 top 속성이 'mainMenuHeight'가 되도록 이동 애니메이션을 수행한다. 또한 step4-4-1 조건과 통합되어 확대 애니메이션도 함께 동시에 진행되어 슬라이딩된다. 이러한 경우는 a-2, b-3, c-3 유형과 같이 초기 시작위치가 메인메뉴리스트 아이템에서 아래로 떨어져 위치하는 경우이므로 메인메뉴리스트 바로 아래까지 이동 애니메이션

을 진행해야 한다.

(step4-4-3) 1단계서브메뉴에 설정한 시작위치 중에서 left속성이 '50%'이거나 '100%'이면서 topFactor가 '0'인 경우에는 left 속성이 '0'이 되도록 이동 애니메이션을 진행한다. 또한, step4-4-1 조건과 통합되어 확대 애니메이션도 함께 동시에 진행한다. topFactor가 '0'이므로 왼쪽으로만 슬라이딩하면 된다. 따라서 left 속성을 '0'으로 설정하여 이동 애니메이션을 진행해야 된다.

(step4-4-4) 1단계서브메뉴에 설정한 시작위치에서 left속성이 '50%'이거나 '100%'이면서 topFactor가 '0.5 '이거나 '1'인 경우에는 메인메뉴리스트로부터 가로, 세로 방향으로 떨어져 위치해 있는 경우이므로 left 속성은 '0', to p 속성은 'mainMenuHeight' 만큼 이동 애니메이션을 진행한다. 또한, step4-4-1 조건과 통합하여 확대 애니메이션도 함께 동시에 진행된다. 이와 같이 step4-4-1에서 step4-4-4의 조건에 해당되는 처리를 수행함으로써 제안한 애니메이션 유형에 대한 확대 및 이동 애니메이션은 완료된다.

[step5] 메인메뉴리스트에 mouseleave 이벤트가 발생하면 이 단계에서는 진행되었던 상태를 초기상태로 환원하는 축소 및 이동 애니메이션에 의한 슬라이딩 처리가 진행된다. 애니메이션 처리가 완료되면 visibility 속성도 'hidden'으로 변경하여 mouseenter 이벤트가 발생하기 이전의 상태로 되돌아가고 알고리즘은 종료된다.

5. 2단계이상서브메뉴의 슬라이딩 알고리즘

서브메뉴의 깊이가 달라져도 애니메이션 알고리즘의 원리는 동일하게 적용된다. 1단계서브메뉴는 메인메뉴리스트를 기준으로 위치되는 반면에 2단계이상서브메뉴는 바로 직전 서브메뉴의 리스트아이템을 기준으로 우측에 배치되므로, 시작위치는 애니메이션 유형에 따른 오프셋만 지정하면 된다. [그림 4]와 [그림 5]는 1단계서브메뉴와 2단계서브메뉴에 각각 b-1 유형과 c-1유형을 적용하여 애니메이션을 진행한 코드로써, [그림 4]는 각 속성들을 설정한 부분적 CSS이고, [그림 5]는 슬라이딩을 진행하기 위해 이벤트를 처리하는 jQuery부분코드이다. 여기에서처럼 위에서 설명한 대로 2단계서브메뉴에 적용된 c-1 유형에 따른 오프셋을 지정하기 위해 .ul-desc-submenu 엘리먼트의 left 속성은 '150%'([그림 4])로, startTopPosition은 [그림 5]에서처럼 'subMenuHeight*topFactor(=0.5)'로 지정하면 된다.

정리하면, 1단계서브메뉴와 2단계이상서브메뉴의 시작위치만 제안한 유형과 같이 설정하면 확대/축소 애니

```
.li-has-child-submenu, .li-has-desc-submenu {
   position: relative;
}
.ul-submenu {
   visibility: hidden;
   position: absolute;
}
.ul-child-submenu { // 1단계서브메뉴
   left: 0;
   height: 0; width : 0;
}
.ul-desc-submenu { // 2단계이상서브메뉴
   left: 150%;
   width: 0; height: 0;
}
```

그림 4. 서브메뉴의 초기 속성 설정

```
                :
mainMenuHeight = $(".ul-topmenu").height();
$("li").has(">.ul-submenu").on({
   mouseenter: function(e) {
                :
      $this = $(this);
      $subMenu = $this.find(".ul-submenu").first();
      subMenuHeight=각서브메뉴리스트아이템수*1개높이;
      if ($this.parent().is(".ul-topmenu")) { // 1단계서브메뉴 처리
         topFactor = 0;
         startTopPosition =
            mainMenuHeight + subMenuHeight * topFactor;
         $subMenu.css({top: startTopPosition});
         $subMenu
            .css({visibility: "visible"})
            .stop(true, false)
            .animate({   // (b-1) 유형 애니메이션 처리
               height: subMenuHeight,
               width: "100%"
            }, "fast");
      } else { // 2단계서브메뉴 처리
         topFactor = 0.5;
         startTopPosition = subMenuHeight * topFactor;
         $subMenu.css({top: startTopPosition});
         $subMenu
            .css({visibility: "visible"})
            .stop(true, false)
            .animate({   // (c-1) 유형 애니메이션 처리
               height: subMenuHeight,
               width: "100%",
               left: "100%",
               top: 0
            }, "fast");
      }
   }, mouseleave: function(e) { // mouseleave 처리
      // mouseleave시 축소/이동 애니메이션 처리가 진행되고,
      // 애니메이션완료 시, visibility 속성을 'hidden'으로변경.
   }
});
```

그림 5. 1, 2단계서브메뉴에 b-1, c-1 애니메이션 유형이 적용된 경우의 이벤트처리

메이션 알고리즘과 확장알고리즘은 서브메뉴의 형태나 깊이에 제한이 없으며, 웹을 기반으로 하는 인터랙티브 콘텐츠를 구현하는 경우에도 쉽게 적용 가능하다.

Ⅳ. 구현 및 실험사례

1. 풀다운메뉴 구현 및 실험

본 논문에서 제안한 방향제어가 가능한 애니메이션 기법을 구현하기 위해 먼저, 엘리먼트 상태나 배치와 관련된 부분은 CSS에서 설정하고 이벤트에 의한 애니메이션 처리는 jQuery를 사용하여 데이터와 동작을 구분하여 유지보수 측면을 고려하여 구현하였다. 또한, 메인메뉴리스트나 서브메뉴가 추가되어도 동작할 수 있도록 확장성을 고려하였다. 풀다운메뉴의 레이아웃은 [그림 6]과 같이 수평메인메뉴 바에 마우스를 올리면 서브메뉴가 아래로 펼쳐지는 일반적인 구조이다. 제안한 애니메이션 기법에 대한 실험에 집중하기 위하여 풀다운메뉴의 기본적인 동작만을 구현하였다. 특별히 서브메뉴의 깊이의 제한은 없으나 여기서는 4단계 서브메뉴까지 구현하여 mouseenter/mouseleave 이벤트에 대한 서브메뉴의 펼침과 접힘 슬라이딩 처리에 대하여 제안한 알고리즘을 모든 서브메뉴에 각 유형을 동일하게 적용하거나, 서브메뉴별로 다른 유형을 적용하여 실험한 결과, 모두 만족할 만한 결과를 확인하였다. 특히, 기존의 풀다운메뉴에서 사용한 상하방식의 단조로운 슬라이딩 기법과 비교하여 사용자요구에 적합한 다양

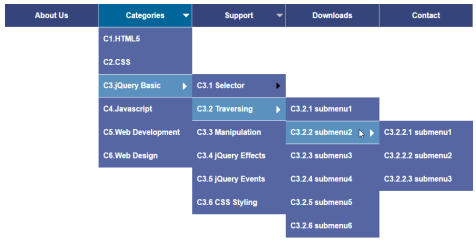

그림 6. 풀다운메뉴의 레이아웃

한 방향의 애니메이션을 사용한 서브메뉴 슬라이딩 처리가 가능해짐에 따라 시각적인 효과가 향상된 결과를 확인하였다. 또한, 서브메뉴가 놓이는 시작위치를 사용자요구에 맞게 변경하는 것만으로도 다양한 애니메이션이 가능하기 때문에 그 효과는 훨씬 더 클 것으로 분석된다.

[표 1]에서는 기존의 상하방식으로 서브메뉴를 슬라이딩한 알고리즘[12]과 제안한 알고리즘을 비교한 결과를 나타낸다. 기존의 방법에서는 지정된 메서드를 사용하여 height에 대한 슬라이딩처리를 수행한 것에 비하여 제안한 방법은 다양한 방향과 형태로 애니메이션 할 수 있기 때문에 성능적인 측면에서 더 우수함을 확인하였다. 특히, 제안한 방법은 웹 기반 인터랙티브 콘텐츠 제작을 위해 쉽게 적용할 수 있는 장점이 있기 때문에 그 응용 범위 또한 클 것으로 사료된다.

2. 제안한 알고리즘을 적용한 이미지슬라이더 구현실험 사례

이절에서는 제안한 기법이 풀다운메뉴에서 뿐만 아

표 1. 알고리즘 비교 분석

항 목	기존방법[12]을 이용한 서브메뉴 애니메이션	제안한 방법을 이용한 서브메뉴 애니메이션
애니메이션에 사용된 기법	.slideDown()/.slideUp() 메서드.	CSS속성과 방향제어가 가능한 확대 및 이동 애니메이션을 이용한 슬라이딩 알고리즘.
애니메이션 방향제어 및 동작	상하방향 슬라이딩 가능. 방향제어 불가능	다양한 방향으로 제어 및 슬라이딩이 가능함.
확대 및 이동 애니메이션	상하방향으로 확대 애니메이션은 부분적으로 가능하나, width/height 속성 제어 및 이동 애니메이션을 할 수 없다.	width/height 속성 및 이동과 관련된 속성 제어가 가능하여 확대 및 이동 애니메이션을 동시에 진행할 수 있기 때문에 다양한 방향과 형태로 슬라이딩이 가능하다.
웹 기반 콘텐츠 적용 가능성	상하방향 슬라이딩 기능만 가능하나 height/width 속성 제어가 어려워 실제로 웹 기반 인터랙티브 콘텐츠제작을 위한 적용 및 응용이 아주 제한적이다.	다양한 웹 기반 인터랙티브 콘텐츠제작에 적용 및 응용이 가능하다.
장단점 분석	jQuery에서 제공하는 .slideDown()/.slideUp() 메서드에만 의존하고 있어서, 사용하기는 간단하나 애니메이션 방향이 상하로 고정되어 있기 때문에, 시각적인 효과를 향상할 수 없고, 커스터마이즈한 애니메이션 기능도 구현하기 어렵다.	CSS속성과 확대 및 이동 애니메이션 기법을 조합하여 슬라이딩처리를 하기 때문에 다양한 형태와 방향으로 서브메뉴를 애니메이션 할 수 있다. 서브메뉴 뿐만 아니라 다양한 웹 기반 콘텐츠 제작에도 쉽게 응용할 수 있다.

니라, 웹 기반 사용자 인터페이스 구현을 위해 적용가능하고 또한 유효한지를 확인하기 위하여 반응형 이미지 슬라이더(Responsive Image Slider)에 제안한 슬라이딩 알고리즘을 적용하는 실험을 수행하였다. 적용실험을 위한 이미지 슬라이더의 HTML은 [그림 1]에서 서브메뉴리스트아이템 li 엘리먼트 안에 ' '와 같은 형태로 태그를 구성하여 이미지를 삽입하였다. 여기에 제안한 알고리즘을 적용하기 위해서는 이미지 자체를 확대 및 축소하는 것이 아니라, 슬라이딩으로 확대 및 축소 대상이 되는 엘리먼트를 포함하는 엘리먼트를 제어해야 하므로 이미지 슬라이더에서는 이미지를 포함하고 있는 div 엘리먼트를 제어하면 된다. 이렇게 구성된 HTML 엘리먼트에 대하여 알고리즘에서 제시한 조건과 절차를 동일하게 적용하였다. 이미지 슬라이더의 스크린영역은 1개의 이미지 크기와 동일하게 980×380으로 구성하였고, 5개의 이미지를 수평방향으로 나열한 이미지전체를 슬라이딩함으로써 이미지가 순서대로 스크린영역에 출력되는 동작을 반복하는 구조를 하고 있다.

이렇게 구현된 이미지 슬라이더에 제안한 기법을 적용하기 위해, 먼저 스크린영역으로 슬라이딩되어 들어올 다음 이미지를 포함하는 엘리먼트(div)에 제안한 유형에 따른 애니메이션을 진행할 시작위치와 알고리즘에서 명시한 초기조건을 미리 설정한다. 이러한 설정에 의해 이미지를 포함하는 엘리먼트는 알고리즘에서 기술한대로 'hidden'상태가 된다. 이미지 슬라이더의 전체 이미지의 슬라이딩이 진행되면, 이러한 진행과 거의 동시에 유형에 따라 조건이 설정된 이미지를 포함하는 엘리먼트가 확대 및 이동 애니메이션으로 슬라이딩되어 스크린영역에 출력된다. 그리고 다시 좌측으로 슬라이딩되어 보이지 않게 된다. 이 때에는 풀다운메뉴에서처럼 축소애니메이션은 필요하지 않다. 이 엘리먼트가 다시 스크린영역에 슬라이딩으로 출력되기 바로 직전에 확대되어진 엘리먼트를 제안한 알고리즘에서 제시한 애니메이션 유형에 따른 속성(height, width, left, top)을 초기화하여 스크린영역으로 슬라이딩되어 들어오는 순간에는 다시 확대 및 이동 애니메이션이 가능해지도록 한다. 이러한 반복적인 절차를 이미지 슬라이더에

그림 7. 제안한 알고리즘을 적용한 이미지 슬라이더

적용하는 실험을 수행하였다. [그림 7]은 적용실험을 위해 제작한 이미지슬라이더로서 제안한 유형 중에서 a-1, b-4, c-1 유형에 대해 애니메이션이 진행 중인 상태를 캡처한 스냅 샷이다. 첫 번째 스냅 샷은 애니메이션 방향이 위에서 아래로 슬라이딩이 진행되고 있는 상태이고, 두 번째 스냅 샷의 중간에 화살표가 그려져 있는 이미지가 대각선 방향으로 슬라이딩이 진행되고 있는 이미지이며, 세 번째 스냅 샷은 이동 애니메이션으로 슬라이딩 되면서 대각선 방향으로 확대가 진행되고 있는 스냅 샷이다. 이러한 진행으로 이미지 슬라이더의 기본 슬라이딩 동작에 본 논문에서 제안한 기법을 적용하는 실험을 진행한 결과, 풀다운메뉴의 서브메뉴에서 애니메이션 처리와 동일한 출력결과를 이미지 슬라이더에서 확인함으로써 풀다운메뉴의 서브메뉴처리에서뿐만 아니라 이미지 슬라이더와 같은 웹 기반 콘텐츠 구현에서도 제안한 기법의 유효성과 적용가능성을 확인하였다. 또한, 이러한 적용실험을 통하여 제안한 기법을 적용한 이미지 슬라이더가 기존의 이미지 슬라이더의 슬라이딩방식에 비해 시각적이고 역동적인 효과를 향상할 수 있음도 함께 확인하였다. 결론적으로 본 논문에서 제안한 방향제어가 가능한 확대/축소 및 이동

애니메이션을 이용한 슬라이딩 기법이 시각적 효과 향상과 아울러 웹 사용자 인터페이스 향상을 위한 새로운 커스터마이즈한 기법임을 확인하였다.

V. 결 론

본 논문은 웹 사용자 인터페이스 향상을 위한 풀다운 메뉴에서 방향제어가 가능한 애니메이션 기법을 제안하였다. 이 기법을 이용하여 풀다운메뉴에서 서브메뉴 펼침과 접힘 처리에 적용한 결과, 기존의 단조로운 단일방향의 서브메뉴처리와 비교하여 확연히 차별화되며, 또한 시각적인 효과를 향상할 수 있다. 이를 위해 본 논문에서는 방향제어가 가능한 확대/축소 애니메이션 기법을 이용한 슬라이딩 알고리즘과 이동 애니메이션을 이용한 확장된 슬라이딩 기법을 확립하여 풀다운 메뉴에서 뿐만 아니라 웹을 기반으로 하는 콘텐츠 제작에 적용할 수 있도록 하였다. 또한 제안한 기법의 유효성과 확장성을 확인하기 위해 풀다운메뉴와 반응형 이미지 슬라이더에 제안한 방법을 적용한 결과 만족할 만한 결과를 도출하였다.

참 고 문 헌

[1] 조세형, “웹사이트 정보품질 및 사용자인터페이스품질이 문화상품 온라인구매의도에 미치는 영향: 고객관여도의 조절효과,” 한국콘텐츠학회논문지, Vol.13, No.12, pp.931-944, 2013.

[2] 최은석, 정승호, 김대용, “WEB2.0 기반 디자인 아이디어 발상 시스템의 사용자 인터페이스 개선,” 한국콘텐츠학회논문지, Vol.10, No.1, pp.37-45, 2010.

[3] J. Park and J. Kim, “Contextual navigation aids for two World Wide Web systems,” International Journal of Human-Computer Interaction, Vol.12, No.2, pp.193-217, 2000.

[4] B. Yoo and J. Kim, “Experiment on the effectiveness of link structure for convenient cybershopping,” Journal of Organization Computing and Electronic Commerce, Vol.10, No.4, pp.241-256, 2000.

[5] 유병민, “인터넷 정보탐색 과정에서 정보구조와 메뉴디자인의 상호작용 분석,” 정보처리학회논문지, 제12-B권, 제4호, pp.473-478, 2005.

[6] 서종환, 김순덕, “네비게이션 디자인에 있어 성별에 따른 선호 스타일 연구,” 감성과학, Vol.8, No.3, pp.221-229, 2005.

[7] 배윤선, 이현주, “유니버설 디자인 개념의 웹 네비게이션 디자인에 관한 연구,” 디자인학연구, 통권 제66호, Vol.19, No.4, pp.101-110, 2006.

[8] 이중엽, “장애인을 위한 유니버설 웹 유저 인터페이 디자인에 관한 연구,” 한국 컴퓨터정보학회논문지, 제12권, 제4호, pp.191-200, 2007.

[9] 이중엽, “유비쿼터스 컴퓨팅 환경에서 노인을 위한 유저 인터페이스 디자인방법 연구,” 한국인터넷방송통신TV학회논문지, 제9권, 제6호, pp.245-251, 2009.

[10] http://www.w3.org/Style/CSS

[11] http://www.jquery.com

[12] 조한수, “하이라이트이미지 애니메이션과 계층적 쌓임 구조를 이용한 풀다운메뉴 설계,” 한국정보기술학회논문지, Vol.14, No.6, pp.167-177, 2016.

저 자 소 개

조 한 수(Han-Soo Cho) 정회원

- 1999년 4월 : 法政대학 공학연구과(박사)
- 2000년 3월 ~ 현재 : 한중대학교 공학부 모바일IT공학전공 부교수

<관심분야> : 패턴인식, 컴퓨터비전, 모바일 및 웹 정보처리