효율적인 CAPTCHA 실험을 위한 웹 기반 실험 환경 구축

Web-Based Experiment Enviornment Implementation for Efficiently CAPTCHA Experiment

정우근
Chung Woo-Keun
부산대학교 컴퓨터공학과
wkchung@pusan.ac.kr

ABSTRACT

우리는 전 보고서 [1]을 통하여 이미지 기반의 CAPTCHA 시스템을 개발하였으며, 보고서[2],[3] 들을 통하여 다각형 기반의 서브 이미지를 형태를 제안하였다. 또한 보고서[4]를 통하여 새로운 서브 이미지 추출방법을 제안하였으며, 사용자 실험을 통하여 효과적인 다각형을 찾았었다. 하지만 이와 같은 보고서들을 전부 오프라인 실험을 통하여 이루어졌으며, 다양한 사용자 계층을 확보하지 못하여 실험에 어려움을 초래하였다. 또한 시스템의 조그만한 변경사항이 있으면 항상 프로그램을 업그레이드해야 하는 불편함이 있었다. 본 단락에서는 효과적인 CAPTCHA 시스템 실험을 제공하기 위하여 웹을 통한 실험 환경을 제안한다. 웹을 통한 CAPTCHA 시스템 환경은 PHP와 Java Applet을 통하여 구현되었다.

KEYWORDS Sub-Image, CAPTCHA, Java Applet, PHP

1 서론

우리는 전 보고서 [1]을 통하여 새로운 이미지 기반의 CAPTCHA 시스템을 제안하였다. 제안된 시스템은 일반적으로 촬영된 사진에서 서브 이미지를 사각형의 형태로 추출한다. 추출된 서브 이미지를 사각형이 가지는 4 가지의 방향 (0, 90, 180 그리고 270 도) 중 한가지를 임의적으로 선택하여 무작위 회전한다. 무작위 회전된 서브 이미지를 올바르게 교정하여 사용자를 인증하는 것이다. 또한 우리는 사용자에게 좀 더 효과적인 서브 이미지를 제공하기 위하여 25 가지 색상 테이블을 이용하여 효과적인 서브 이미지를 제공하기 위하여 필터링 시스템을 제공하였다[5]. 우리는 여기서 일반적인 이미지에서 사각형의 형태로 출력하였으나, 사용자가 어떤 형태의 서브 이미지가 가장 효율적인지 판단하기 위하여 보고서 [2], [3]을 통하여 사용자에게 가장 효율적인 다각형을 실험을 통하여 알아보았다. 실험에 사용된 다각형은 정사각형, 정오각형, 정육각형, 정칠각형, 정팔각형이 사용되었다. 지금까지 구현된 시스템들은 일반적인 이미지에서 서브 이미지를 추출할 시 원 이미지의 비율에 비례하여 서브 이미지를 크기를 설정하여 서브 이미지를 추출하였다. 하지만 보고서 [4] 에서는 지금 까지와는 다른 방법을 통하여 서브 이미지를 추출 및 실험을 행하였다. 새롭게 제안된 서브 이미지 추출방법은 사용자가 원의 반지름을 정한 뒤, 선택된 반지름의 크기를 가지는 원으로 서브 이미지를 추출, 추출된 서브 이미지에서 다각형의 형태로 서브 이미지를 추출하였다.

하지만 이와 같이 구현된 시스템들의 실험 환경은 전부 오프라인에서 이루어졌다. 오프라인으로 실험할 경우 다양한 사용자의 확보가 힘들고, 시스템의 업그레이드시 매번 환경을 재설정해주어야 하는 번거로움이 따른다. 그리하여 본 보고서에서는 웹을 통한 실험 환경을 제공하여, 보다 더 높은 사용자를 확보할 수 있는 환경을 제공한다. 먼저 구현된 시스템의 사용방법을 알아 보자.

2 구현된 시스템

본 단락에서는 웹을 통하여 구현된 CAPTCHA 시스템의 환경을 간단히 살펴보고 Java Applet 내에서 DB를 연동하는 방법과 Applet 내에서 이미지 로드에 대해서 간략히 살펴보도록 하겠다.

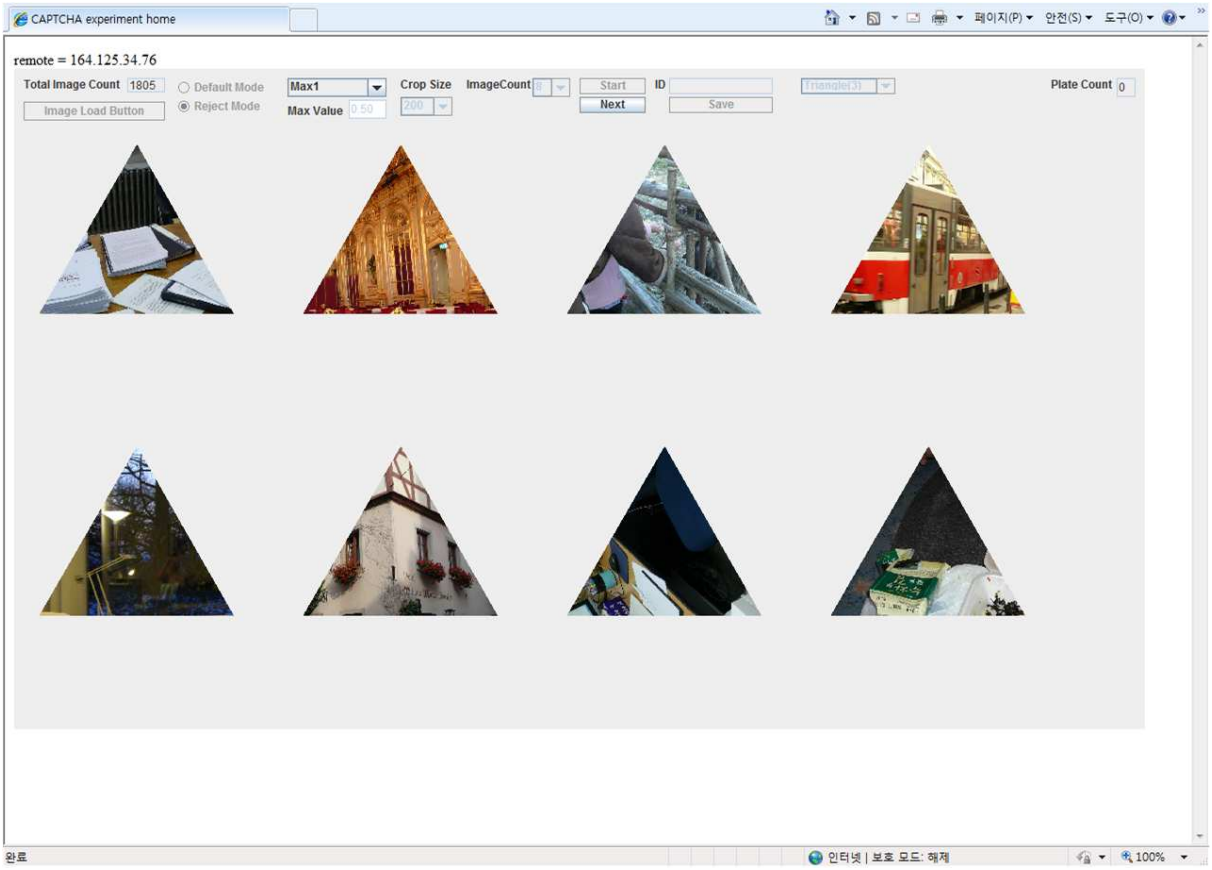

그림 1. 본 보고서에서 소개할 CAPTCHA 시스템의 실험을 웹에서 실험할 수 있도록 구현한 시스템이다. 기존에 제공되는 사용 방법은 기존에 사용되었던 오프라인 버전과 동일하다.

그림 1은 본 보고서에서 제공하는 CAPTCHA 시스템 실험 환경이다. 외관적인 구조와 기본적인 메카니즘은 오프라인에서 제공되었던 또는 전 보고서 에서 제공되었던 시스템과 동일한 시스템이다. 전 보고서와 동일하게 서브 이미지의 추출방법은 원의 넓이에 근거하여 추출하였다. 하지만 전 보고서에서는 제공되었던 시스템은 사용자가 임했던 CAPTCHA 시스템의 실험값들을 전부 파일로 저장하여, 저장된 파일들을 파싱을 통하여 실험값을 분석하였다. 실험 환경을 오프라인으로 제공하였기 때문이다. 하지만 본 보고서에서는 이런 불편을 해결하기 위하여 DB에 실험값들을 저장하여 보다 쉽게 실험값들을 분석하고 저장할 수 있다. DB에 저장되는 테이블의 구조와 웹에서 접근하는 메카니즘에 대하여 알아보도록 하겠다.

본 보고서에서 제공된 시스템은 Neobio 즉, 리눅스 환경에 설치되어있으며, 기본적인 URL 주소는 http://neobio.cs.pusan.ac.kr/~captcha 이다. 본 웹 기반의 CAPTCHA 시스템 실험의 환경은 Java Applet으로 구현되었다. Java Applet으로 구현할 시, 집적적인 이미지로드에 대한 문제가 발생한다. 웹 상에서 이미지를 로드하기 때문에 설치되어 있는 로칼 호스트 경로를 통하여 이미지 로드가 불가능 하기 때문이다. 즉, 모든 파일 입력이 URL 주소를 통하여 입력되는 것이다. 하지만 Java Applet에 있는 이미지 로드 기본 API 함수인 public Image getImage(URL url, String name) 을 통하여 이미지를 로드할 수 있다. getImage 함수는 이미지가 상주해있는

Field	Type	Null	Key	Default	Extra
userID	varchar(40)	NO	PRI		
resultID	int(11)	NO	PRI	NULL	auto_increment
expDate	text	NO			
remoteIP	text	NO			
numberOfGame	int(11)	NO			
accuracy	double	NO			
plateCount	int(11)	NO			

표 1. 웹에서 CAPTCHA 실험에 임한 사용자들의 실험값을 효율적으로 저장하기 위한 User-Inform 테이블의 구조

URL 주소를 기입하고, 해당 이미지 파일명을 name에 기입하면 이미지가 로드 되어 Image 변수를 반환한다. 예를 든다면 http://neobio.cs.pusan.ac.kr/~captcha/1.jpg 를 로드 한다면, URL에 http://neobio.cs.pusan.ac.kr/~captcha 를 기입하고, name에 1.jpg를 기입하면 Image가 로드되는 것이다. URL 주소는 Applet이 설치되어있는 PC가 가지는 URL을 함수 getDocumentBase() 를 통하여 쉽게 가져올 수 있다. 이와 같은 함수는 Applet 함수이므로 Applet 내에서만 실행이 가능하다. 하지만 이렇게 Applet 클래스에 존재하는 getImage() 함수만 사용한다면 Applet 클래스에서 모든 것을 해결해야 할 것이다. 하지만 이와 같이 URL 주소를 통하여 이미지를 로드할 수 있는 함수는 Toolkit 클래스에도 존재한다. Toolkit에 존재하는 이미지 로드 함수는 Applet에서 제공하는 이미지 로드 함수와 같은 파라메터 값을 쓰고, 반환 값도 Image로써 동일한 함수이다. Applet에서 getDocumentBase()를 통하여 URL를 저장하여 직접적인 이미지를 로그하고 처리하는 클래스에서 Toolkit getImage(URL url, String name) 와 함께 쓴다면 별 무리 없이 이미지를 로드할 수 있을 것이다.

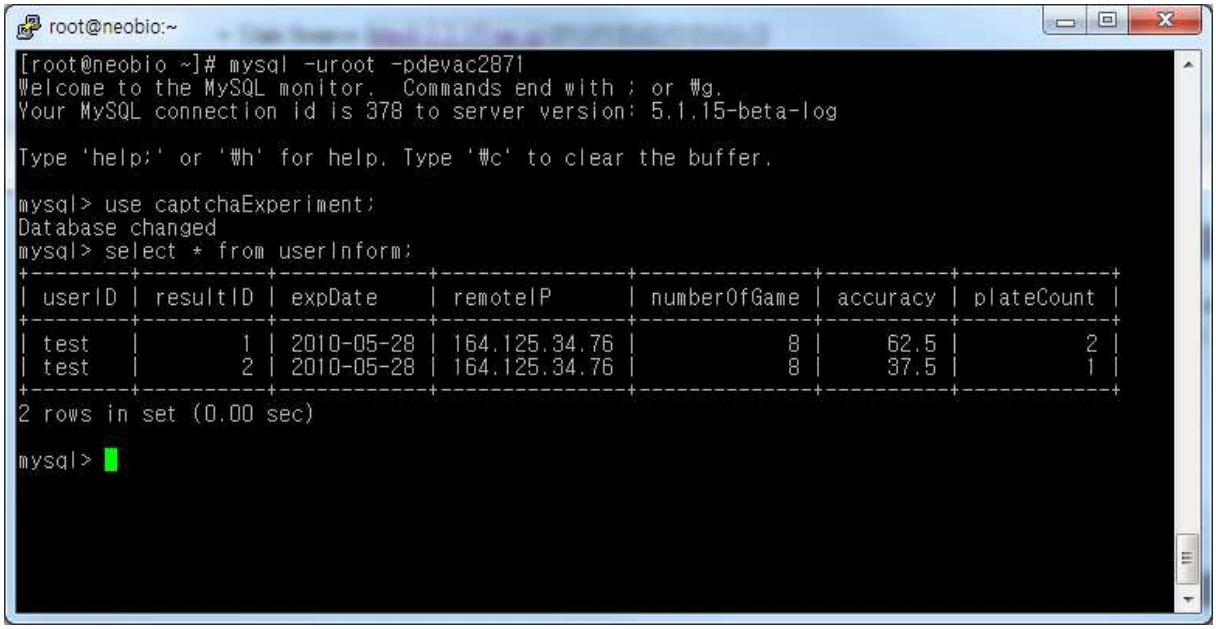

그림 2. 그림을 살펴보면 동일한 userID에 따라서 다른 resultID가 생성된 것을 알 수 있다.

본 보고서에서 제공하는 CAPTCHA 시스템은 최대 8장의 이미지를 로드하여, 로드된 이미지에서 서브이미지를 추출하기 때문에 비교적 큰 메모리가 할당되어야 한다. 하지만 Applet은 웹 브라우저에서 실행되기 때문이 따로 실행 옵션 값을 따로 줄 수가 없다. 기본적으로 Java의 옵션값으로 -Xmx(메모리량)m으로 설정할

수 있다. 하지만 이와 같은 옵션 값은 Applet을 로드하는 곳에서 간단한 옵션 값으로 메모리를 설정할 수 있다.

```
<PARAM name=java_arguments value="-Xmx1024m">
```

위의 소스를 살펴보자. 위의 소스는 Windows Explorer 브라우저를 사용하여 Applet을 불러오는 코드중 일부분이다. Applet을 불러온 뒤, 파라메터 값을 지정할 때, 다음과 같이 파라메터 값을 지정하면, 1기가의 Hash Memory를 보유하는 Applet을 사용할 수 있다.

Field	Type	Null	Key	Default	Extra
userID	varchar(40)	NO	PRI		
resultID	int(11)	NO			
fileName	text	NO			
cropSize	int(11)	NO			
regionType	int(11)	NO			
rejectType	text	NO			
rejectValue	float	NO			
imgWidth	int(11)	NO			
imgHeight	int(11)	NO			
cropImgWidth	int(11)	NO			
cropImgHeight	int(11)	NO			
pointX	float	NO			
pointY	float	NO			
reSample	int(11)	NO			
mouseClicked	int(11)	NO			
max1	float	NO			
..	..	..			
max25	float	NO			
answer	text	NO			

표 2. 웹에서 CAPTCHA 실험에 임한 사용자들의 실험값을 효율적으로 저장하기 위한 resultOfExp 테이블의 구조

본 웹 환경의 CAPTCHA 시스템은 사용자가 임했던 실험값들을 DB에 저장한다. Applet 코드가 아닌 기본적인 Java 코드에서는 Java 홈페이지에서 자신이 선택한 DB에 맞는 DB-Connector.jar 파일을 다운받아서 Java가 설치되어 있는 Lib폴더 내에 DB-Connector.jar 파일이 존재한다면 Java 코드내에서 DB에 관련된 작업을 수행할 수 있을 것이다. 하지만 Applet 내에서 DB와 관련된 작업을 수행하는 것은 환경 값이 틀리다. Applet은 로드 또는 수행하는 당시에 모든 환경은 자신이 설치되어 있는 로컬 컴퓨터의 URL 주소에서 수행이 되기 때문에 Java가 설치되어 있는 로컬 컴퓨터 폴더내에 DB-connector.jar가 있다고 하더라도 Applet은 DB-Connector에 대한 오류를 발생시킬뿐이다. Applet 내에서 DB를 사용할려면 DB-Connector.jar 파일이 Applet 파일이 존재하는 곳에 상주해야 하며, Applet 로드 당시에 파라메터 값으로 파일 명을 넣어주면 Applet 내에서 DB를 무난히 사용할 수 있따. 아래에 코드를 살펴보자. 아래는 Applet 을 로드하는 부분인데 archieve 속성에 DB-connector.jar 파일명을 명시해주면 Applet 내에서 DB 관련된 코드를 사용할 수 있다.

```
<applet code="MApplet.class" archive='mysql-connector-java-5.0.3-bin.jar' width=1200 height=700 >
```

본 웹 환경의 CAPTCHA 시스템에서는 사용자가 실험했던 실험값들을 DB에 저장한다고 하였다. 실험값들 비교분석하기 위해서는 사용자가 임했던 실험값들을 효율적으로 넣기위해서 효율적인 테이블이 필요하다. 표 1는 사용자가 임했던 실험값들에 대하여 사용자의 정보에 대해서 저장하는 표이다. 표를 살펴보자. 표 1은 총 7의 속성으로 이루어져있다. userID의 속성은 CAPTCHA 실험에 임하는 사용자의 ID 값을 저장시키는 값이며, resultID는 동일한 사용자가 계속해서 실험할 경우, 각각의 실험값을 구분지을 수 있는 ID 값에 해당하며, 사용자마다 자동생성하게 해주어 동일한 사용자 또는 다른 사용자들간에 각각의 다른 값들을 지정되어 쉽게 구분될 수 있다. test라는 유저 아이디로 2번의 실험을 통하여 결과 값을 통해 좀더 면밀히 살펴보도록 하자. 그림 2을 살펴보면 동일한 userID에 따러서 resultID가 각각 다르게 생성된 것을 알수가 있다. 이것은 동일한 userID를 가진 사용자가 2번의 실험에 임했다는 것이다. 이 resultID는 각각의 userID에 따라서 고유하게 생성된다. 또한 각각의 사용자마다 remoteIP를 기록하고 있으며, accuracy 속성을 통하여 사용자의 성공률을 간단ㅇ하게 파악할 수 있다. plateCount는 사용자가 실험에 임한 횟수를 나타내고 있다. 이 userInform 을 통하여 우리는 실험에 임한 사용자의 성공률과 실험 횟수를 간단하게 알 수 있다.

효과적인 실험분석을 위하여 사용자 관련 정보를 제외한 실험값들을 저장하는 테이블을 구성하였다. 테이블의 구조는 표 2 와 같다. ResultOfExp의 테이블은 많은 속성 값들을 가지고있다. 사용자가 실험을 끝내고 데이터를 저장할 때 userInform에 아이디와 고유의 인덱스 값이 생성된다. 이 아이디 값과 고유의 인덱스 값으로 resultOfExp에 나머지 실험 환경의 데이터를 저장하는 것이다. 실험값을 분석할 때는 이 테이블을 이용하여 각 사진에 대한 성공률, 사진이 가지는 색상분포도, 사용자가 각 사진에 대한 교정 횟수등이 저장된다.

3 결론

지금까지 우리는 이미지 기반의 CAPTCHA 시스템을 개발해왔다. 우리는 효율적인 서브 이미지를 제공 또는 각 도형, 서브 이미지의 크기에 따른 사용자의 성공율을 알아 보기 위하여 실험을 행하였다. 하지만 현재까지 개발해온 CAPTCHA 시스템은 오프라인의 시스템이라서 실험값들에 대한 분석 및 자료 수집이 번거로웠었다. 그리하여 본 보고서에서는 효과적인 실험 환경과 실험값에 대한 분석을 보다 쉽고 간편하게 하기위하여 웹 환경의 CAPTCHA 시스템 실험 환경을 구축하였다. 웹 환경은 Java Applet을 Linux 환경에 구축하였으며, 사용자가 임한 실험값에 대한 저장은 MySQL을 통하여 데이터를 저장하였다. 본 보고서를 통하여 효과적인 실험 환경을 구축하였으며, 추후 과제로써는 실험값에 대한 분석을 보다 간편하게 하기 위한 간단한 사용자 분석 시스템을 하는 것이다. 본 웹 환경을 통하여 보다 많은 사용자를 확보하여 더 많은 실험을 통하여 좀 더 효과적인 시스템을 구축하고 실험값을 통하여 CAPTCHA 환경을 개선해야 할 것이다.

참고 문헌

1. 정우근, “실험을 통한 효과적인 captcha 시스템을 위한 서브 이미지 추출 개선방안,” 부산대학교, Tech. Rep., 11 2009.
2. ——, “효과적인 서브 이미지 기반의 captcha 시스템을 위한 다각형 형태의 서브 이미지 실험,” 부산대학교, Tech. Rep., 01 2010.
3. ——, “다각형 넓이를 이용한 captcha 시스템의 사용자 인식률 실험,” 부산대학교, Tech. Rep., 02 2010.
4. ——, “원의 넓이에 근거한 새로운 서브 이미지 추출방법을 통한 captcha 시스템 개발 및 사용자 인식률 실험,” 부산대학교, Tech. Rep., 03 2010.
5. ——, “Color 색상 분포도 실험을 통한 효과적인 서브 이미지 제공,” 부산대학교, Tech. Rep., 12 2009.