문서에 그림 넣기

Karnes

2008 년 9 월 23 일

요 약

이 글은 KC2008 테스트 과정에서 알게 된 사실을 요약하는 것이다. KC2008 을 설치하면 TEXLive 2008 을 텍시스템으로 사용하게 되는데, 이전의 외부 그림 넣기 방법과 달라진 점이 몇 가지 있어 이를 적어두려 한다.

PSTricks 나 MetaPost 및 PGF/tikz 와 같은 이른바 "그림 그리기 언어"와 관련된 부분은 거론하지 않는다. 이 글의 목적은 오직 외부에서 이미 제작된 그림을 어떻게 문서에 삽입하느냐에 관한 것이다.

이 글의 주장은 필자의 개인적인 생각이며, 이것을 받아들이는 것은 전적으로 사용자의 몫이다. 이 주장이 "옳다"고 강변할 생각은 없다.

차 례

1 먼저 몇 가지

다음과 같이 몇 가지를 미리 제시해두고자 한다.

1. 외부 그림 넣기는 매우 드라이버-의존적(driver-dependent)이다. 그러므로 예컨대 dvipdfmx라는 드라이버를 타겟으로 하여 작성된 문서는 dvips나 pdflatex으로 컴파일해서 결과를 얻을 수 없다.

   a) TEXLive 2008에 와서 그림 처리에 있어 드라이버 의존성은 더욱 심화되었다. 예전에는 웬만하면 그냥 넘어갔기 때문에 예컨대 dvips 옵션을 주고 문서를 작성해도 dvipdfmx로 컴파일하는 것이 그다지 어렵지 않았다. 그러나 이제는 거의 불가능해졌다고 생각하면 된다.

   b) dvipdfm와 dvipdfmx는 전혀 다른 드라이버이다. 이 점은 매우 중요한데, dvipdfmx가 독자적인 드라이버 정의 파일을 갖지 못했을 당시 dvipdfm의 것을 빌어다가 쓰는 경향이 있어 dvipdfm과 dvipdfmx를 거의 동일시하는 관행이 굳어졌으나, 적어도 그림 넣기에 관한 한 이 둘은 전혀 다르게 동작하며, 최종적으로 얻어지는 결과도 다르다.

2. 드라이버마다 사용할 수 있는 그림 포맷이 다르다.

   a) dvips는 `.eps` 이외의 포맷 그림을 전혀 알지 못한다. 그리고 최종출력물을 얻을 때 dvips는 사용하지 않을 것이므로 `.eps` 그림도 더이상 문제삼지 않기로 한다.[1)]

   b) dvi 출력 포맷은 더이상 사용하지 않는다. 그러므로 이따금 `.wmf`나 `.bmp` 등을 화면에 보여주곤 하던 dvi driver가 있기는 했어도 이 프로그램군을 더이상 사용하지 않고 dvipdfmx와 pdflatex만 고려하기로 할 때 이와 같은 그림 포맷은 사용하지 않아야 할 것이다.

   c) dvipdfmx는 (`.eps`), `.png`, `.jpg`, `.pdf` 포맷의 그림을 잘 처리한다.

   d) pdflatex은 `.png`, `.jpg`, `.mps`, `.pdf` 포맷의 그림을 잘 처리한다.

   e) 결국 사용할 수 있는 그림 포맷은 이 공통부분, 다시 말해 `.png`, `.jpg`, `.pdf` 만을 문제삼는 것이 현명하다고 판단한다. `.png`와 `.jpg`는 주로 래스터(비트맵) 그림에, 그리고 벡터 그림은 `.pdf`로 작성하는 것이 좋다.

1) 그런데, KC2008의 시대에 와서 dvips는 오직 단 한 가지 목적, 즉 pstricks 언어로 작성된 그림을 컴파일하기 위한 목적밖에는 쓸 데가 없다. 또한, 이미 pstricks를 그림 언어로 채택했다면 외부 그림을 넣을 일은 아주 드물 것이다. 결과적으로, 이제 "`.eps` 그림 포맷은 더이상 사용하지 않는다"고 말해도 틀리지 않다고 판단한다.

이상을 요약하면 다음과 같다. 텍 작업의 최종출력물은 어떤 경로를 거치든 pdf이다. pstricks를 사용하는 경우를 제외하면 dvipdfmx나 pdflatex이 pdf를 얻는 주요한 루트가 된다. dvips는 pstricks 그림이 포함된 문서를 작성할 때만 사용한다. 그러므로 외부 그림은 pdf나 png(jpg)로 작성해야 한다. 또한 그림 포함을 위해서는 특정 루트를 타겟으로 해야 하며 이 타겟이 아닌 다른 문서 빌드 루트를 따르게 되면 오류가 발생한다.

2 드라이버별 그림 넣기 위한 설정

2.1 pdflatex에서 그림 넣기

pdflatex은 특히 pdf 그림에 강하다. 그러나 eps 그림을 전혀 사용할 수 없으며, ps 계열 중에서는 오직 `.mps` 즉 MetaPost PS만을 포함할 수 있다.[2)]

```
\usepackage[pdftex]{graphicx}
\usepackage[pdftex,svgnames]{xcolor}
```

이 두 행으로 pdflatex을 실행하기 위해 필요한 조치는 다 한 셈이다. 다만 이 설정으로는 latex으로 컴파일할 수 없다. 즉 dvips/dvipdfmx 어떤 방법으로도 pdf 결과를 얻어낼 수 없을 것이다. `[pdftex]` 옵션은, pdflatex이 실행되고 있다면 반드시 지정해야 할 필요는 없다. 그러나 이 옵션이 있으면 설령 latex 명령을 실행하더라도 pdf 파일이 만들어지므로, pdflatex의 실행을 확정적으로 확보할 수 있다는 장점이 있다.

문서에서 color를 사용하지 않는데 두번째 행이 필요한가 의문일 수 있다. 그러나 pdfcolor라는 내부 명령을 처리하여야 pdf 그림을 불러올 수 있는데 이 때문에 xcolor 패키지가 필요하다. 즉, (경우에 따라) xcolor를 로드하지 않으면 pdf 그림을 불러오지 못할 수 있다. xcolor의 옵션 `svgnames`는 예시로서, 꼭 이 설정을 지정해야 하는 것은 아니다.

2.2 dvipdfmx를 위한 그림 넣기 설정

dvipdfmx를 위한 설정은 다음과 같다.

```
\usepackage[dvipdfmx]{graphicx}
```

2) 이곳저곳에, epstopdf 패키지를 로드하면 eps 그림도 불러올 수 있다고 하는 것을 볼 수 있다. 이 말은 사실이지만, 처음부터 pdf로 만들어진 그림과 eps에서 만들어진 pdf는 경우에 따라 동일하지 않다. 그림 그리기 도구에서 직접 pdf로 export한 것이 당연히 가장 품질이 좋다. eps에서 pdf로 변환하면서 ghostscript를 거치는데, 반드시 동일성이 보장된다고는 하기 어렵다. 따라서 epstopdf와 같은 잠정적인, 궁여지책의 패키지는 쓰지 않는 것이 옳을 것이다. 그러나 epstopdf 유틸리티가 상당히 신뢰할 만한 변환 툴인 것만은 사실이다.

이 경우에는 xcolor를 꼭 로드할 필요는 없지만, 안전하게 하려면 그것도 적어주는 편이 낫다.

```
\usepackage[dvipdfmx,svgnames]{xcolor}
```

latex을 실행할 때 반드시 `--shell` 옵션을 지정하도록 하라. 그래야만 에러없이 진행되고 그림의 크기를 `dvipdfmx`에게 알려줄 수 있다. KCmenu 또는 KCmenu를 탑재한 Notepad++와 같이 이 작업이 자동화되어 있는 유틸리티를 쓰면 세세한 데 신경쓸 필요가 없겠지만.

dvipdfmx는 eps 그림을 포함하여, pdf, png, jpg 등의 그림을 잘 처리한다. 다만 eps 그림은 ghostscript를 이용하여 pdf로 변환하는 것이기 때문에 역시 처음 그림의 동일성이 보장되지 않는다. 그러므로 되도록 eps 그림을 쓰지 않는 것이 좋을 것이다.

특히, dvipdfmx를 통하여 pgf/tikz를 사용하려 한다면 이 드라이버의 명시적 선언이 반드시 필요하다.

2.3 dvips를 위한 그림 넣기 설정

현재 dvips는 매우 특수한 목적에 한정되어서 쓰인다. 그 특수한 목적이란 다름아닌 pstricks의 활용이다. 어쨌든, pstricks는 dvips에서밖에 제대로 처리되지 않기 때문이다. pdftricks니, pst-pdf니 하는 것들도 알고보면 전부 dvips를 통하여 그림을 처리한 후에 그것을 pdf로 바꾸어서 불러오는 데 불과하다.

dvips는 한때 '표준' 취급을 받았던 eps 이외의 그림을 전혀 처리할 수 없다. 그러므로 dvips를 기본 드라이버로 사용하려 한다면 모든 png, jpg, pdf 그림을 모두 eps로 변환해야 한다. 그런데, eps는 지금까지 수많은 쟁점을 불러왔다. 여기에는 두 가지 문제가 있는데, 하나는 프로그램마다 만들어내는 eps의 양식이, ps 프린터의 종류만큼 많아서 통일되어 있지 않다는 것이다. 예컨대 Adobe Photoshop이 (기본값으로) 만들어낸 eps는 Adobe Photoshop에서밖에 제대로 읽어지지 않는다고 해도 좋을 정도이다. MS Visio와 같은 프로그램도 다른 프로그램과 호환되지 않는 eps를 만드는 것으로 유명했다. 두번째는 Postscript 언어 표준을 정한 Adobe사가 ps 언어의 개발을 중지했다는 것이다. 혼란상을 그대로 두고 손을 떼어버렸기 때문에 eps 및 ps 그림은 속수무책의 상황에 빠져 있다. 우리는 ghostscript가 잘 처리하고 dvips가 해석하는 ps가 표준이기를 바라지만, 그게 그렇게 만만치 않은 것이다. 이 말은, dvips가 해석하지 못하거나 잘못 해석하는 ps 또는 eps 파일이 얼마든지 있을 수 있다는 것이다.

그래서, imagemagick이나 ghostscript 유틸리티는 가끔 벡터 포맷의 eps를 래스터 eps로 변환하기도 한다. 그림이 비슷하게 읽어지도록 하는 데는 좋지만 결국 그림의 품질은 희생하게 된다.

그러므로, dvips를 사용할 때는 전략적인 접근이 필요하다. 처음부터 clean한 eps 그림을 준비하는 데 많은 공을 들여야 한다. Photoshop으로 eps를 만들지 않아야 하며, 한때 유용한 솔루션처럼 보였던 wmf를 eps로 변환하는 따위의 일은 되도록 해서 안 된다. 또한, 많은 양의 pstricks 그림이 있는 문서가 아니라면, 즉 한두 개의 pstricks 그림을 포함하려 한다면 이 그림들을 별도의 pdf로 미리 제작해두고 pdflatex이나 dvipdfmx를 통하는 것이 훨씬 나은 결과를 가져올 수 있다.

아무튼지,

```
\usepackage[dvips]{graphicx}
```

이렇게 드라이버를 명시하면 나머지 문제는 pdftex 엔진과 dvips가 할 일이다. dvips 옵션이 선언되면, pdflatex이나 dvipdfmx로는 절대로 결과를 얻지 못하게 되므로 특히 주의해야 한다.

2.4 둘 이상의 드라이버에 호환되도록 코드를 작성

가끔 KTUG의 고수들이 작성하는 코드를 보면,

```
\ifpdf
  \usepackage[pdftex]{graphicx,xcolor}
\else
  \usepackage[dvipdfmx]{graphicx,xcolor}
\fi
```

와 같은 내용을 볼 수 있다. 이것은 pdflatex이나 dvipdfmx 두 가지 경우 모두 동작하도록 하기 위한 것이다. 그러나 이것은 latex으로도 컴파일해보고 pdflatex으로도 해보고 하는 테스트용 파일을 작성하는 데 쓰거나, 여러 드라이버 옵션을 받아들여 동작하게 하기 위한 패키지의 작성에나 쓸 것으로, 일반적인 문서에서는 그다지 좋은 모양은 아니라고 생각한다.

3 결론

지금까지 한 얘기를 요약하자.

1. 문서작성시에 타겟 드라이버를 미리 설정하고 이것 하나만 명시적으로 사용한다. 일반적으로 pdflatex이 표준으로 권장되지만 pdf 파일의 크기를 줄이고 싶거나 pdf 파일에 암호를 걸고 싶거나 하여 dvipdfmx를 쓰는 것도 좋다. 어쨌든 자신이 선택한 한 가지 방법을 일관되게 사용할 것이며, 문서의 첫머리에 이 사실을 적어두는 것이 좋다. 또한, 어떤 패키지는 pdflatex에서만 동작하거나 하

는 등 일부 차이점도 있으므로 그 사실을 명확히 파악해서 문서를 작성해야 할 것이다.

2. eps 그림은 되도록 피한다.

3. dvips 사용을 되도록 피한다. pstricks를 사용하는 목적이 postscript 언어를 이용하여 확장된 기능을 사용하려는 것이 아니라면, pgf나 다른 대안 언어를 찾아보는 것도 좋다.[3] 문서에 삽입되는 pstricks 그림이 열 개 안팎이라면, 각각의 그림을 별도로 제작하는 방법도 생각해보라.

그림의 질은 문서 품위의 절반 이상을 결정하는 것이다. 주의깊게 선택하여 훌륭한 문서를 만드는 데 집중하도록 하자. Happy TEX'ing.

3) 그러나 최근 pstricks의 눈부신 발전을 무시할 수도 없다. pstricks는 정말 훌륭한 언어라고 생각한다.