Some TEXMACS Image Inclusion Considerations

BY DAVID E. MILLER

Email: david.miller@quaoar.us
Web: mathboxvm.org

15 May 2013

The author is DAVID E. MILLER. He is a graduate of the UNIVERSITY OF CINCINNATI with a BS degree in Aerospace Engineering and a graduate of THE OHIO STATE UNIVERSITY with an MS degree in Systems Engineering. He lives in PICKERINGTON, OHIO.

Abstract

This article elaborates on the issues involved with the inclusion (inserting or linking) of images within a TEXMACS document. It includes a recommended work-flow for the inclusion of images (or graphics) of various formats that circumvents the complications and idiosyncrasies inherent with TEXMACS image inclusion. Included are some suggested helper tools and information related to images and formats. This article is not a comprehensive treatment of all aspects of TEXMACS image inclusion. Comments and recommendations are welcome.

1 Introduction

This article consists of an elaboration of notes made while investigating the means used by TEXMACS for inserting images. These notes are not a comprehensive treatment of all the issues that may be involved with the general subject of images and formats, or the specifics of the technical matter of insertion of images for inclusion within TEXMACS documents. This author was interested in including images created by or exported from other software tools. In particular, the images exported from GEOGEBRA and GNUPLOT (which may have been created using the MAXIMA `plot2d` function or the `draw` packages were of interest. In the case of MAXIMA, images might also have been created using a TEXMACS MAXIMA session as an interface by way of the built-in plugin. The means used to create image files is actually irrelevant to the general technical points made in this article. However, this information is referenced for the sake of providing the reader with the details of the motivating context which may be important to some.

2 Points of Emphasis

There are issues involved with the inclusion (linking or inserting) of images in a TEXMACS document. The following list is a summary of some significant points for emphasis:

- TEXMACS has two primary means for including images — *linking* and *insertion*. By inserting an image file, the image becomes an integral part of the TEXMACS document. Subsequent changes to the image file do not affect the inserted image content. A linked image is a reference to the image file itself which is not part of the TEXMACS document. Subsequent changes to the referenced image file should be reflected by the image content of the TEXMACS document.

- TEXMACS converts image content to POSTSCRIPT format regardless of the format of the image file itself. This is accomplished by "helper" programs external to TEXMACS. The particular helper program used varies according to the image format. Obviously the required programs must be accessible on the system in use.

- While it is possible, within the limitations of the available helper programs, to convert and insert images of various supported formats using the built-in features of TEXMACS, this is not necessarily the best way, in general, to accomplish image inclusion. This author recommends that images be created as or converted to PDF files, converted to ENCAPSULATED POSTSCRIPT (EPS) format, and then inserted into or linked to the TEXMACS document. There are many tools available for creating PDF image files or for converting other image formats to PDF format. GEOGEBRA exports to PDF directly, and GNUPLOT (or MAXIMA using GNUPLOT) has the PDFCAIRO terminal used to create PDF image files, as examples.

- PDF files used as a source for conversion and inclusion within a TEXMACS document should first be created with dimensions (in inches, centimeters, points etc.) that are close to the intended dimensions of the image within the document. TEXMACS can scale images of larger dimensions, but if the actual dimensions of the images are much larger than the dimensions intended to be used within the document, the TEXMACS document file size will be larger than necessary. If this is not an issue then the PDF image dimensions are not significant — assuming the PDF file has been cropped of excessive white space as required. Image dimensions smaller than that intended may not produce

acceptable results if scaled by TEXMACS to larger dimensions. Linked images also do not significantly affect the size of the TEXMACS document.

- Programs that create or export images to PDF format may produce image files that are entire pages (A4, 8.5 by 11 inches, etc.) in dimensions with the image of the intended dimensions placed somewhere on the page. These PDF files in general must be cropped to remove this unnecessary and excessive white space. Once cropped the PDF files may be converted to EPS for insertion in, or linking to, a TEXMACS document.

- Images that include so-called *transparency* effects pose unique issues. PostScript has limited or no support for transparency effects[1] of various forms — invisible pixels, opacity, translucency, etc. PDF images with transparency must be *rasterized* in order to display as intended. This has implications for source image dimensions, resolution, and file size. Rasterized PDF files in the form of EPS created to accommodate transparency effects can be much larger than the source PDF file — typically 20 times larger or more depending on the resolution used in the conversion to EPS.

- PostScript images inserted in or linked to TEXMACS may appear to render with poor quality when viewed using the TEXMACS program itself. This is not something that should worry the user. Images should render as expected if the document is exported to PDF — or printed.

- TEXMACS documents intended for export to PDF will either be intended for *printing* or for *display*, or perhaps both. TEXMACS documents exported to PDF intended for printing may require higher image resolutions when converting PDF image files to EPS for inclusion.

- TEXMACS documents that include a large number of EPS images of rasterized PDF files with transparency effects may result in very large file sizes if these images are inserted instead of linked. In these cases linking images is recommended. Note that a document exported by TEXMACS as PDF will result in a PDF file that is significantly smaller in size than might be expected if the document involves large EPS image file sizes due to image insertion or linking.

- PDF files exported by TEXMACS for printing may be "re-sized" or otherwise processed by external tools after being exported by TEXMACS to achieve various results, such as for display using a browser or other device. PDF files for display purposes do not require the higher resolutions that printing may demand. Re-sizing PDF files after exporting with resolutions necessary for printing circumvents the need to have different versions of images depending on the intended purpose — printing or display. Note again that resolution has a significant effect on the file size of EPS images converted by rasterizing PDF files with transparency effects.

1. The PostScript language has limited support for *full* (not "partial") transparency effects. For more detailed technical information about PDF and PostScript language support for image transparency effects refer to: `http://en.wikipedia.org/wiki/Transparency_%28graphic%29#Transparency_in_PDF`

3 Recommended Work-flow For Image Inclusion

This author recommends the following work-flow for the inclusion (linking or inserting) of images within TEXMACS documents:

1. Create, export or otherwise convert all images to PDF format using smallest dimensions consistent with the intended size in the TEXMACS document. As examples:
   - GNUPLOT using `pdfcairo` terminal to PDF files
   - MAXIMA `plot` functions or `draw` package to PDF files
   - GEOGEBRA export to PDF files
   - Conversions using `a2ping` (discussed below) to PDF files
   - ADOBE ACROBAT — Convert to ABOBE PDF

2. Crop all PDF files with images as required to remove excessive white space using:

```
pdfcrop <infilename>.pdf <outfilename>.pdf
```

3. Convert all PDF image files from PDF to EPS using:

```
pdftops -level3 -eps -r 300 <infilename>.pdf <outfilename>.eps
```

   where `-level3` sets the POSTSCRIPT level and `-r 300` sets the resolution — `300` is the default. The resolution should match the TEXMACS Printer dpi setting. If the TEXMACS file will be exported to PDF the intended purpose (printing, display, or both) will determine the resolution. Note that PDF files exported by TEXMACS will likely have relatively smaller file sizes compared to the TEXMACS source documents with inserted images.

4. Link or insert the EPS image for inclusion within the TEXMACS document using:

   Insert→Image→Small figure or Insert→Image→Large figure menu selections.

   then

   Insert→Link Image... or Insert→Insert image... menu selections

   Note. Linking is probably the best option in general. An exception is if the TEXMACS document itself is being distributed.

5. Scale the image using TEXMACS Width and Height settings if required. Refer to the TEXMACS Help menu selection for details about image scaling options.
6. Use File→Export→Pdf... menu selection to export the TEXMACS document to PDF
7. Use the following command to reduce the file size of exported PDF files:

```
gs -sDEVICE=pdfwrite -dCompatibilityLevel=1.4 -dPDFSETTINGS=/screen
-dNOPAUSE -dQUIET -dBATCH -sOutputFile=<outfilename>.pdf <infilename>.pdf
```

   `PDFSETTINGS` options are:

   - `/screen` (screen-view-only quality, 72 dpi images)
   - `/ebook` (low quality, 150 dpi images)
   - `/printer` (high quality, 300 dpi images)
   - `/prepress` (high quality, color preserving, 300 dpi images)

- `/default` (almost identical to `/screen`)

Thus, a single source PDF file exported from a TEXMACS document with a page `dpi` setting of `300` or more and with EPS images inserted with resolutions of `300` or more may be "re-sized" according requirements — print or display.

Note. Depending on the `PDFSETTINGS` option used, the following message may appear when using this command:

```
GPL Ghostscript 9.06: Set UseCIEColor for UseDeviceIndependentColor to work
properly.
```

Adding the following option to the `gs` command will prevent this message from appearing:

```
-dUseCIEColor
```

Paper size selection options:

- `-sPAPERSIZE=letter`
- `-sPAPERSIZE=a4`
- `-dDEVICEWIDTHPOINTS=w`
- `-dDEVICEHEIGHTPOINTS=h` (point=1/72 of an inch)
- `-dFIXEDMEDIA` (force paper size over the PostScript defined size)

Other options:

- `-dEmbedAllFonts=true`
- `-dSubsetFonts=false`
- `-dFirstPage=pagenumber`
- `-dLastPage=pagenumber`
- `-dAutoRotatePages=/PageByPage`
- `-dAutoRotatePages=/All`
- `-dAutoRotatePages=/None`
- `-r1200` (resolution for pattern fills and fonts converted to bitmaps)
- `-sPDFPassword=password`

Note. This above recommended work-flow assumes that:

– the TeX Live TEX auxiliary programs (texlive-extra-utils) package is installed (`pdfcrop`)[2] and
– the PDF utilities (poppler-utils) package (based on Poppler[3]) is installed (`pdftops`) and
– the Ghostscript package (`gs`) is installed.

4 Converting Other Image Formats To PDF

Converting images to PDF (e.g., PNG, JPG, etc.) may be accomplished by available tools for that purpose. If the TeX Live TEX auxiliary programs (see Note above) and the sam2p packages are installed, then the following may be used to convert image files to PDF for step 1 above:

2. There is also a pdfcrop project: http://pdfcrop.sourceforge.net/. This is not the program referenced here.
3. Poppler is a PDF rendering library based on Xpdf PDF viewer

```
a2ping -v --hires <infilename>.[png|jpg|tif|etc.] <outfilename>.pdf
```

Note. `a2ping.pl` is a Perl script that calls on external helper programs to do its work. Input formats include: PS (PostScript), EPS, PDF, PNG, JPEG, TIFF, PNM, BMP, GIF, LBM, XPM, PCX, TGA. Use `man a2ping` for more details. The `a2ping` tool has numerous options for converting files.

The following code is a shell script (named `pdf2ps`) written by Herbert Voss that may be used or modified according to individual needs. This shell script accomplishes steps 2 and 3 of the recommended work-flow described above. It could be modified to include the above command for converting images to PDF with the result being an EPS file ready for inclusion in a TEXMACS document. This author has not used this shell script.

```
#!/bin/sh
# $Id: pdf2eps,v 0.01 2005/10/28 00:55:46 Herbert Voss Exp $
# Convert PDF to encapsulated PostScript.
# usage:
# pdf2eps <page number> <pdf file without ext>

pdfcrop "$2.pdf" "$2-temp.pdf"
pdftops -f $1 -l $1 -eps "$2-temp.pdf" "$2.eps"
rm "$2-temp.pdf"
```

5 Some Sample Images

The following figures where inserted in or linked to the TEXMACS document used to create this article as a PDF file. These are included for demonstration purposes only and do not represent sophisticated or comprehensive examples. They are useful for the purpose at hand nonetheless.

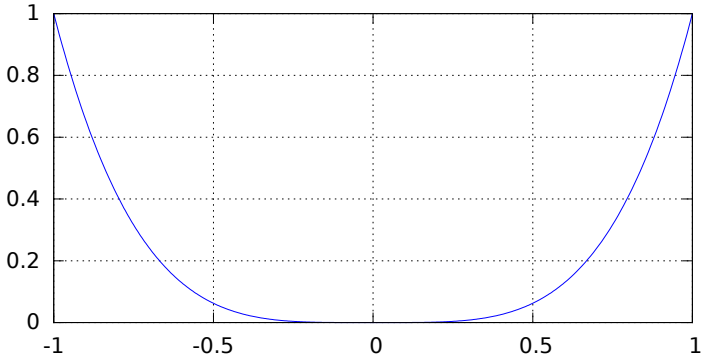

Figure 1. EPS image created using GNUPLOT and the `epscairo color` terminal

Figure 1. is an inserted EPS image. It was created using GNUPLOT with the `epscairo` terminal. In this case the recommended work-flow is unnecessary because with the `epscairo` terminal set GNUPLOT creates an EPS file directly with no conversion required, and no transparency effects are involved. The `epscairo` terminal has a default "canvas" size of 5 in width by 3 in height. This image was created with the size options set to 4in width by 2 in height. The GNUPLOT statement used for this purpose is:

```
set terminal epscairo enhanced color font "sans,12" size 4in,2in
```

To create this same image file using the Maxima `plot2d` function or the `draw` package requires the user to include these gnuplot options according to the particular methods used by these two means of image creation. Refer to the Maxima Manual for the details.

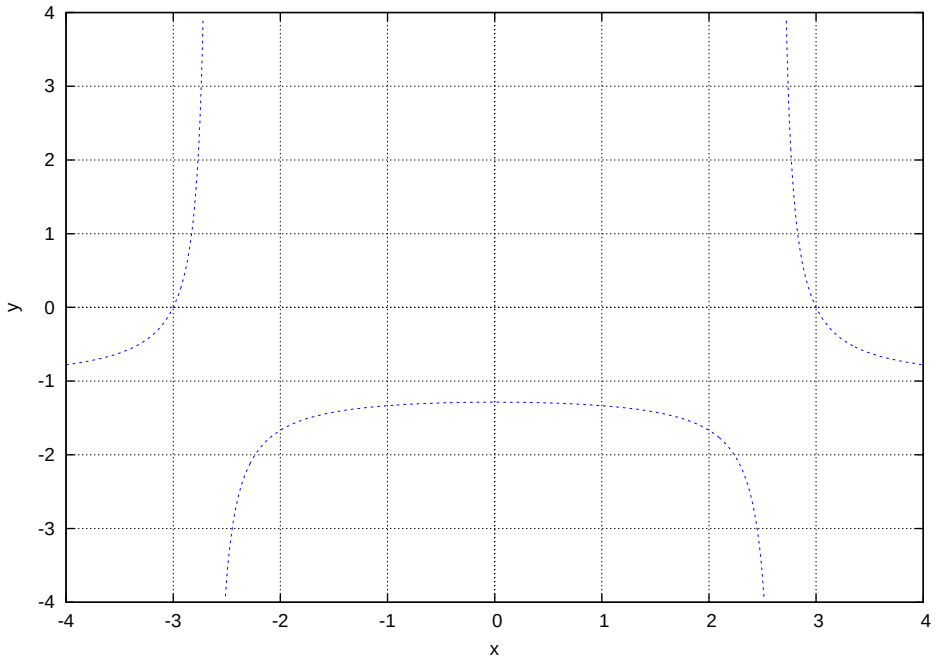

Figure 2. EPS image created using gnuplot and the `postscript eps color` terminal

Figure 2. is an image of a plot generated using gnuplot and the `postscript eps color` terminal. Since no transparency effects are involved, the image created in encapsulated PostScript form is inserted directly into this document using the built-in image insertion menu selection of $\TeX_{\text{MACS}}$. This obviates the need for conversion to PDF.

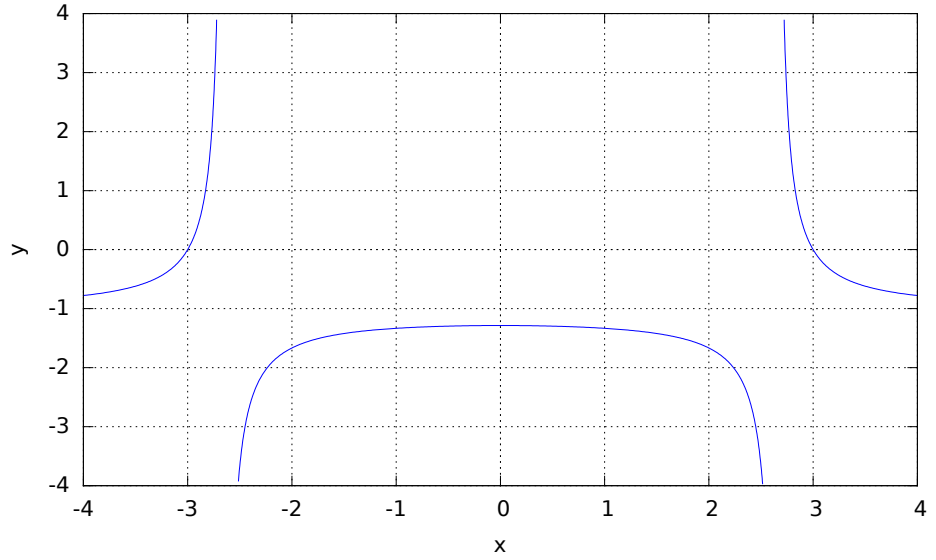

Figure 3. PDF image created using gnuplot and the `pdfcairo` terminal converted to EPS using `pdftops -eps`

Figure 3. is a PDF image of the same plot as Figure 2. created using GNUPLOT. In this case the `pdfcairo` terminal was used to create the PDF image file. This file was converted to EPS using the `pdftops` tool with the `-eps` (ENCAPSULATED POSTSCRIPT) option set. The resulting ENCAPSULATED POSTSCRIPT image is inserted directly into this document using the built-in image insertion menu selection of TEXMACS. Since no transparency is involved the resulting EPS file is not unusually large compared to the parent PDF file.

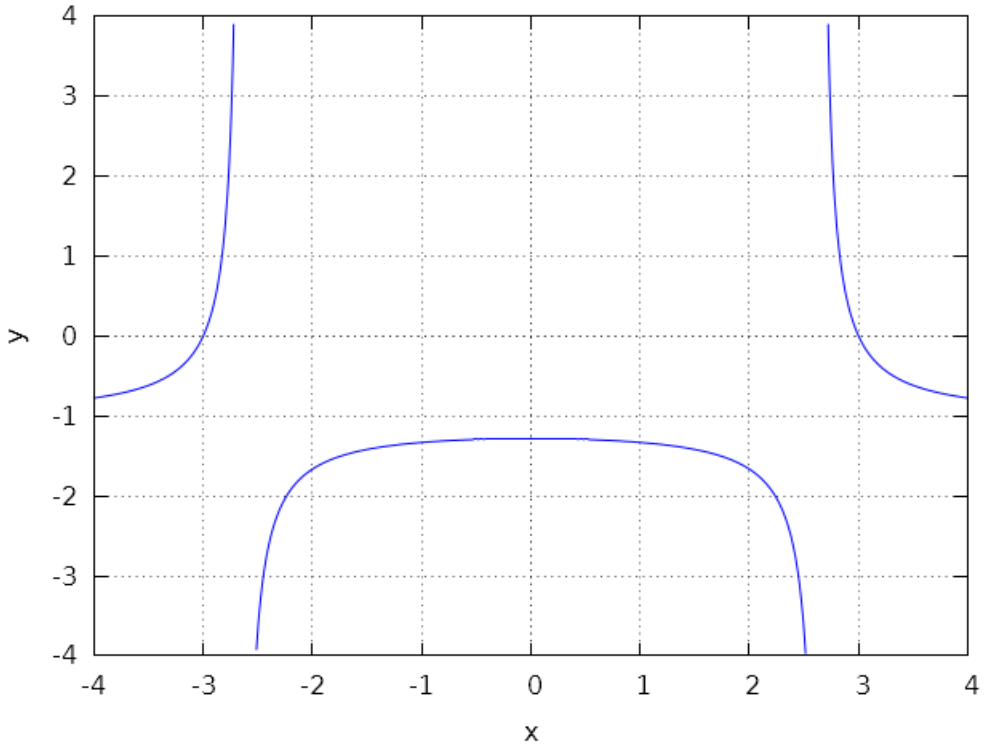

Figure 4. PNG image created using GNUPLOT and the `pngcairo` terminal

The plot shown as Figure 4. is a PNG image file created using GNUPLOT with the `pngcairo` terminal set, and inserted directly using the built-in image insertion menu selection of TEXMACS. Like the images of the figures above, this image does not involve any transparency effects. This obviates the need for conversion to PDF and EPS. The image was created with a resolution of 640 by 480 pixels. When exported to PDF from TEXMACS the image display quality may be marginal. Higher resolution may be required to improve the display or print quality.

Note. The reader should be aware of an anomaly that occurs when exporting from a TEXMACS document that contains PNG images to PDF. The quality of the results will depend on the version of TEXMACS used and the PDF reader program used. This PDF file was exported from TEXMACS from the latest MS WINDOWS version. If this same file is exported to PDF from the GNU/LINUX version of TEXMACS, Figure 4. above will appear in the PDF file rendered using the GNU/LINUX EVINCE document viewer program with very good quality. If this same PDF file is rendered using the ADOBE ACROBAT READER, Figure 4. renders with much poorer quality. EVINCE uses POPPLER/CAIRO whereas ADOBE ACROBAT READER does not. However, if the PDF file is exported from the MS WINDOWS version of TEXMACS this issue does not arise. Figure 4. renders well

in both PDF readers. Obviously, something is happening differently with the PDF export process when using the MS Windows version of TEXMACS. As of the date of this article, the cause of this anomaly is a mystery.

When creating or exporting to PNG images the user must be aware of the effect resolution has on the quality of the image when exported from TEXMACS to a PDF document or printed.. A four inch wide image intended for printing at 300 dpi requires a PNG image with a width of no less than 1200 px. This image is then scaled to 4 in when inserted in or linked to a TEXMACS document. Thus, the dpi is 1200 px ÷ 4 in or 300 px/in. If the resolution is inadequate then the result may be "jaggies" in the images of the exported PDF file.

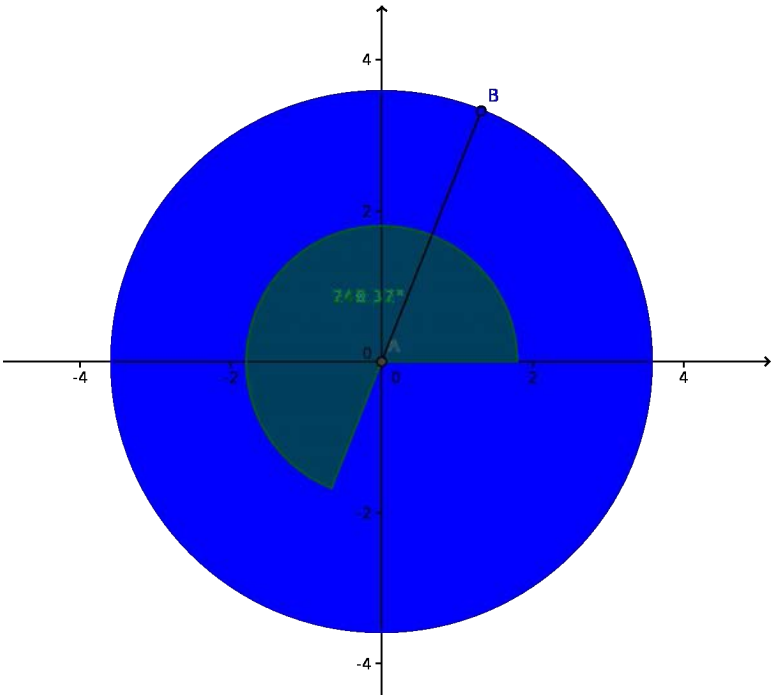

Figure 5. Exported Geogebra PNG image with transparency inserted into TEXMACS without conversion.

Figure 5. above is a PNG image exported from Geogebra and inserted directly into this TEXMACS document. Note that the transparency effects of the original PNG image file are not retained if this image is converted to PDF. While this PNG may render as expected when viewed from the source TEXMACS document, it does not render as expected when exported to PDF

It should be noted that if transparency effects are not involved with the image, then image files created, exported or saved directly as EPS (e.g, gnuplot `epscairo`, Geogebra export to EPS, etc.) are probably the best choices in this case. Compare this image to that of Figure 8. below which is a rasterized PDF file with transparency effects converted to EPS.

Note. For Maxima and TEXMACS users — executing the `plot2d()` function with default options from TEXMACS using the Maxima plugin will cause a *wxt* gnuplot terminal display window to appear displaying the plot. However, the only way to save the plot as a graphic format is to copy it to the clipboard using the features of the *wxt* terminal graphics display window. There is no way to paste a graphics image directly into a TEXMACS document from the clipboard unfortunately. The best alternative is to use the gnuplot options available to write the image directly to a PDF file using the `pdfcairo` terminal setting. This may be accomplished after using the *wxt* terminal display window to verify that the plot is as intended.

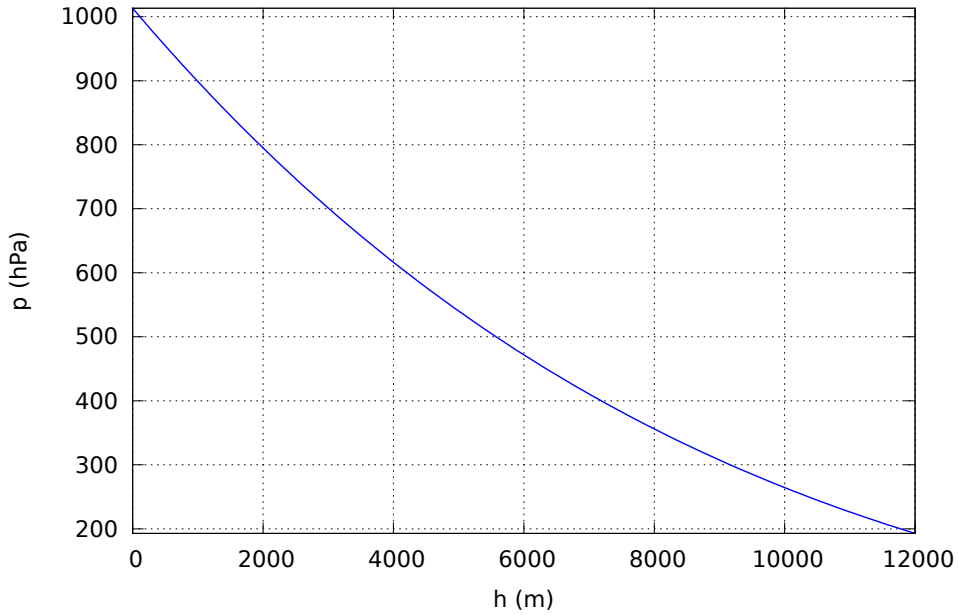

Figure 6. PDF image created using the MAXIMA `draw` package with the `pdfcairo` terminal and `pdftops -eps`

The PDF image of Figure 6. was created using MAXIMA and the `draw` package. The `draw` package is a MAXIMA interface to GNUPLOT, so indirectly the PDF file used for this figure was created using GNUPLOT. The resulting PDF file was converted to EPS using the `pdftops` tool with the `-eps` (ENCAPSULATED POSTSCRIPT) option set. There are no transparency effects involved, so the size of the resulting EPS file is not unusually large compared to the parent PDF file. The EPS file was inserted directly using the built-in image insertion menu selection of TEXMACS.

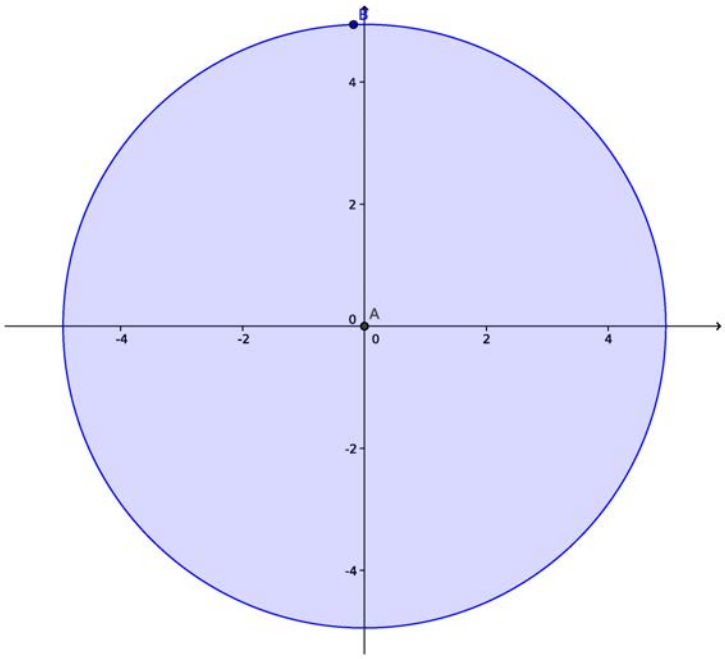

Figure 7. Exported GEOGEBRA PDF image converted to EPS using `pdftops -eps` and inserted into TEXMACS

The PDF file of Figure 7. was created as an exported image using the Geogebra program. This PDF file retains the transparency attributes of the Geogebra objects — in this case a simple circle object that is filled with the same color as the circumference, but with some degree of "opacity" imposed on the fill color. While the PDF file can accommodate this kind of transparency effect if exported from Geogebra, PostScript only has limited support for *full* transparency effect. Therefore an image of this type must be converted to EPS and the opacity fill effect rasterized in order for the intended effect to render properly when the containing TEXMACS document is exported to PDF. This is easy to do using the `pdftops` tool with the `-eps` option as described above in the section for the recommended work-flow. The resulting EPS file is significantly larger than the parent PDF file. The EPS file of Figure 7. was inserted using the built-in image insertion menu selection of TEXMACS. Consequently the file size of the TEXMACS document increased significantly.

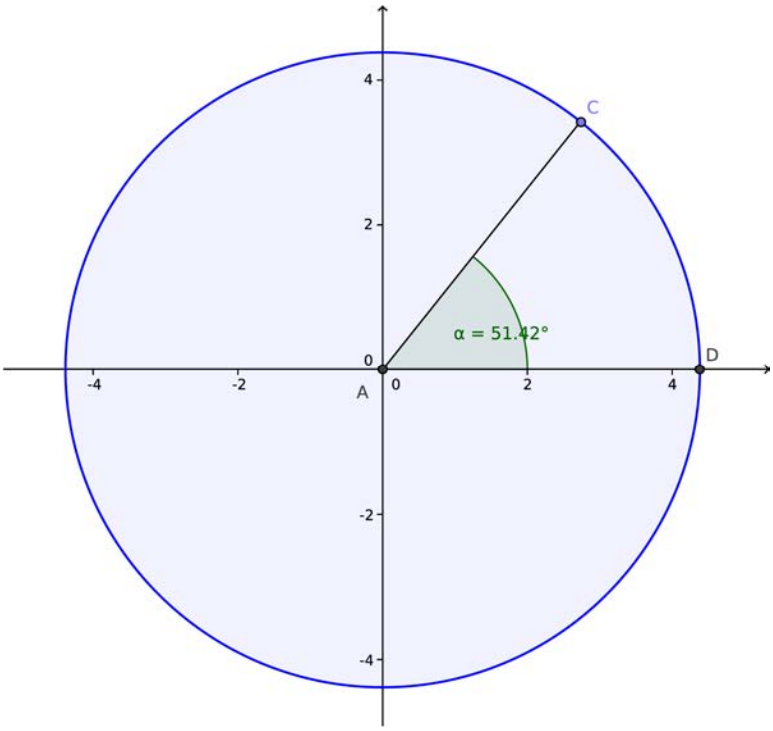

Figure 8. Exported Geogebra PDF image converted to EPS using `pdftops -eps` and linked to TEXMACS

The above image of Figure 8. is also a result of PDF image exported from Geogebra and converted to EPS using `pdftops` tool with the `-eps` option. This image involves slightly more complex transparency effects. This image must also be rasterized during conversion to EPS. For the same reason as that for the EPS image of Figure 7. above, this results in a large EPS image file size relative to the parent PDF file. Due to the large EPS file size this image is linked to the TEXMACS document instead of being inserted. Thus the file size of the TEXMACS document is not significantly affected.

For resolutions above 300 dpi (e.g., `pdftops -r 600`) the size of a file with these dimensions, resolution, and complexity can be 50 MB or more. This significantly larger relative file size cannot be avoided, since it is the inevitable consequence of rasterization. But, it can be mitigated to some degree by creating source PDF files with dimensions (inches, centimeters, points, etc.) as close as possible to the intended dimensions for inclusion in the TEXMACS document.

For example, this image is approximately square with a TEXMACS Width setting of `4in`. Thus, barring any other constraints for the dimensions of the parent PDF file, creating this parent PDF file with dimensions close to 4 inches wide by 4 inches high, for example, will result in a relatively smaller EPS files than intentionally or inadvertently creating a file with larger dimensions. Slightly larger dimensions for PDF files are acceptable since TEXMACS can scale the image to the desired dimensions during inclusion.

Also resulting EPS file sizes are very sensitive to the resolution used during conversion of the parent PDF file. The default resolution of the `pdftops` tool is `300`. Anything over the default increases the EPS file size significantly. On the other hand, using a lower resolution settings than the default when converting PDF to EPS will result in significantly lower EPS file sizes. Resolutions of 150-300 dpi (`pdftops -r 150` to `-r 300`) are reasonable in this context. The most common resolutions used are: 72, 96, 150, 300, and 600. These resolutions should be adequate for most routine purposes.

Note. The following two shell commands will provide all of the information a user should need to know about a PDF file including dimensions:

```
pdfinfo <filename>.pdf

identify -verbose <filename>.pdf
```

The resulting information displayed will show page and other dimensions in *points* — `pts`. A point is 1/72 of an inch. Also, 1 inch is 2.54 centimeters. Therefore, one inch is 72 points, and one centimeter is 28.346457 points. These conversion factors can be useful. For example, GEOGEBRA exports images in PDF format using centimeters. So to export a 4 in wide PDF image using GEOGEBRA requires an export width setting of $4 \cdot 2.54 = 10.16$ cm. However, GEOGEBRA exports this image centered in the middle of a PDF page size that is much larger. To get the PDF image requires cropping of the exported PDF file to get rid of the nuisance white space. Attempting to include the exported PDF file without cropping will require scaling of the width to 4 in and the height of the included image will probably be excessive and the overall appearance of the image will be smaller than expected. Converting the uncropped PDF file to EPS for insertion has the similar consequences, and if transparency effects are involved, the EPS file will be huge.

The implication of this digression into dimensions and conversion factors is that some trial and error may be involved in getting at the right export settings. If excessive file sizes are not an issue, then export the image to PDF with dimensions as large as desired, crop the resulting PDF file, convert it to EPS with 300 or more dpi resolution, link it to TEXMACS and scale the width as necessary or desired. The PDF file exported from TEXMACS will be relatively large, but as noted above, since this file presumably is of high resolution, it can be post-processed using GHOSTSCRIPT (`gs`) to re-size the file as required.

6 Questions Requiring Answers

It may be asked why it is necessary to use this work-flow. Why not merely use the built-in insertion and linking features of TEXMACS with images? There are several reasons to consider:

1. Image files are converted to POSTSCRIPT by TEXMACS using various external helper programs. The user converting images externally allows control over the attributes of the image file that is being included, especially the resolution — in this case EPS as defined by the user using external tools and option settings.
2. PDF images are often created or exported to pages of higher dimensions with excessive white space. Thus, as is they are often too large for insertion to or linking as they are — these may not display or export to PDF as expected. These files should be cropped first. TEXMACS uses the `pdf2ps` external helper program to convert PDF files to POSTSCRIPT. This helper program does not crop PDF files to remove excessive white space. Also, any transparency effects will not be retained.
3. TEXMACS (UBUNTU package v1.0.7.15) crashes when inserting or linking to PDF files. The latest MS WINDOWS version of TEXMACS (v1.0.7.18) does not crash, and will display inserted or linked cropped PDF files correctly, but the included PDF images may be missing if the TEXMACS document is exported to PDF. Such current and future TEXMACS idiosyncrasies are avoided by converting images to EPS prior to inclusion — by insertion or linking.
4. Image files inserted into TEXMACS using built-in features may render for display and export to PDF at acceptable quality unless transparency effects are involved. In this case the recommended work-flow steps above are required in order to preserve the transparency.

5. Considering all the above, using the same method for inclusion of all image files avoids having to deal with different *modus operandi* and TEXMACS conversion idiosyncrasies depending on the format of the image file. The recommended work-flow of this article should work for all image formats with and without transparency effects. The only downside is that is requires a little more effort on the part of the user to get things right with images beforehand, and possibly relatively larger EPS file sizes may be involved. PDF files exported from TEXMACS may be re-sized according to need.

One final point of emphasis. The above assumes that the TEXMACS operating system environment is GNU/Linux or one similar. If the environment is MS Windows or OS X then it may be the case that this is not optimal for using the above recommended work-flow. While it may be an important goal of the TEXMACS developers to have a presence in these operating system environments, it has to be recognized that in practice this presents limitations and constraints that many users will not realize until they run into difficulty when trying to do something as simple as inserting or linking an image to a TEXMACS document using this recommended work-flow. For this reason this author uses a GNU/Linux VirtualBox virtual machine

Acknowledgment

This article was created using GNU TEXMACS. The websites for this project are `http://www.texmacs.org` and `http://www.gnu.org/software/texmacs`.