

Mali™ GPU Texture Compression Tool

Version: 2.2

User Guide

ARM®

Mali GPU Texture Compression Tool

User Guide

Copyright © 2009 ARM. All rights reserved.

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
14 October 2009	A	Non-Confidential	First release for v2.2

Proprietary Notice

Words and logos marked with ® or ™ are registered trademarks or trademarks of ARM® in the EU and other countries, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

Where the term ARM is used it means “ARM or any of its subsidiaries as appropriate”.

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Unrestricted Access is an ARM internal classification.

Product Status

The information in this document is final, that is for a developed product.

Web Address

<http://www.arm.com>

Contents

Mali GPU Texture Compression Tool User Guide

Preface

About this book	x
Feedback	xiii

Chapter 1

Introduction

1.1 About the Mali GPU Texture Compression Tool	1-2
---	-----

Chapter 2

Installing the Texture Compression Tool

2.1 Installing the Texture Compression Tool on Microsoft Windows	2-2
2.2 Installing the Texture Compression Tool on Linux	2-3

Chapter 3

Using the Texture Compression Tool

3.1 Starting the Texture Compression Tool	3-2
3.2 The Texture Compression Tool GUI	3-3
3.3 Opening textures	3-5
3.4 Setting the output directory	3-6
3.5 Compressing textures	3-7
3.6 Using the Texture Compression Tool on the command line	3-10

Glossary

List of Tables

Mali GPU Texture Compression Tool User Guide

	Change history	ii
Table 1-1	File formats the Texture Compression Tool supports	1-2
Table 3-1	Texture Compression Tool GUI buttons	3-4
Table 3-2	Options for ETC compression	3-8

List of Figures

Mali GPU Texture Compression Tool User Guide

Figure 3-1	Texture Compression Tool window	3-3
Figure 3-2	Compression options	3-7
Figure 3-3	Compression results	3-9

Preface

This preface introduces the *Mali GPU Texture Compression Tool*. It contains the following sections:

- *About this book* on page x
- *Feedback* on page xiii.

About this book

This is the *Mali GPU Texture Compression Tool User Guide*. It provides guidelines for using the Mali GPU Texture Compression Tool to assist in the development of applications for Mali *Graphics Processing Units* (GPU). This book is part of a suite belonging to the Mali Developer Tools.

Intended audience

This guide is written for software developers who are writing OpenGL ES or OpenVG applications for a Mali GPU.

Using this book

This book is organized into the following chapters:

Chapter 1 *Introduction*

Read this for an introduction to the Mali GPU Texture Compression Tool.

Chapter 2 *Installing the Texture Compression Tool*

Read this chapter for information about how to install the Texture Compression Tool.

Chapter 3 *Using the Texture Compression Tool*

Read this chapter for information about how to compress images so that they can be used more efficiently on Mali GPUs. The Mali GPU Texture Compression Tool is intended to be used by developers of 2D and 3D content.

Glossary Read this chapter for a description of terms used in this document.

Conventions

Conventions that this book can use are described in:

- *Typographical*

Typographical

The typographical conventions are:

italic Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.

bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
monospace	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.
<u>monospace</u>	Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<i>monospace italic</i>	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
monospace bold	Denotes language keywords when used outside example code.
< and >	Enclose replaceable terms for assembler syntax where they appear in code or code fragments. For example: MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcod _e _2>

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, <http://infocenter.arm.com>, for access to ARM documentation.

ARM publications

This book contains information that is specific to this product. See the following documents for other relevant information:

- *Mali GPU Developer Tools Technical Overview* (ARM DUI 501)
- *Mali GPU Performance Analysis Tool User Guide* (ARM DUI 0502)
- *Mali GPU Shader Developer Studio User Guide* (ARM DUI 0504)
- *Mali GPU Demo Engine User Guide* (ARM DUI 0505)
- *OpenGL ES 1.1 Emulator User Guide* (ARM DUI 0506)
- *Mali GPU Binary Asset Exporter User Guide* (ARM DUI 0507)
- *Mali GPU Shader Library User Guide* (ARM DUI 0510)
- *OpenGL ES 2.0. Emulator User Guide* (ARM DUI 0511)
- *Mali GPU Offline Shader Compiler User Guide* (ARM DUI 0513).

Other publications

This section lists relevant documents published by third parties:

- *OpenGL ES 1.1 Specification* at <http://www.khronos.org>.
- *OpenGL ES 2.0 Specification* at <http://www.khronos.org>.
- *OpenGL ES Shading Language Specification* at <http://www.khronos.org>.
- *OpenVG 1.0 Specification* at <http://www.khronos.org>.
- *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2* (5th Edition, 2005), Addison-Wesley Professional. ISBN 0-321-33573-2.
- *OpenGL Shading Language* (2nd Edition, 2006), Addison-Wesley Professional. ISBN 0-321-33489-2.

Feedback

ARM welcomes feedback on this product and its documentation.

Feedback on this product

If you have any comments or suggestions about this product then contact malidevelopers@arm.com and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to errata@arm.com. Give:

- the title
- the number, DUI 0503A
- the page numbers to which your comments apply
- a concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Chapter 1

Introduction

This chapter describes the Mali GPU Texture Compression Tool. It contains the following section:

- *About the Mali GPU Texture Compression Tool* on page 1-2.

1.1 About the Mali GPU Texture Compression Tool

This section gives an overview of the Mali GPU Texture Compression Tool.

Texture compression enables you to reduce the bandwidth usage required to load textures in graphics applications, this can give your application superior performance and reduces the power consumption of your platform.

The Texture Compression Tool runs on your Windows or Linux workstation, it enables you to compress individual textures or multiple textures. You can view the original texture and the compressed texture together for comparison

Your graphics application running on your platform, reads and displays the compressed texture data produced by the Texture Compression Tool.

The Texture Compression Tool is compatible with Windows XP and Linux, See Chapter 2 *Installing the Texture Compression Tool*.

You can use the Texture Compression Tool to compress textures in a variety of different formats, Table 1-1 shows the file formats that the Texture Compression Tool can read and convert.

Table 1-1 File formats the Texture Compression Tool supports

Format	Description
.bmp	BitMaP format. A format used to store digital images.
.jpeg, .jpg	Joint Photographic Experts Group format. A lossy compressed image format.
.tga	Truevision Graphics Adapter format. A raster graphics file format.
.pbm	Portable BitMap format. A monochrome version of .ppm. A basic level exchange format.
.pgm	Portable GreyMap format. A basic level exchange format.
.png	Portable Network Graphics format. A lossless compressed image format.
.ppm	Portable Pixel Map format. A basic level exchange format.
.xbm	The X BitMap format. An ASCII text monochrome image format used on X Windows.
.xpm	The X PixMap format. An ASCII text based image format used on X Windows.
.pkm	ETC format. Files compressed using ETC texture compression. You can use the Texture Compression Tool to view .pkm files.

The Texture Compression Tool compresses textures using the *Ericsson Texture Compression* (ETC) algorithm. ETC is a 4bpp texture compression algorithm. ETC is the Khronos recommended texture compression algorithm.

The Texture Compression Tool can be used from a *Graphical User Interface* (GUI) or, from a command line, for more information see *Using the Texture Compression Tool on the command line* on page 3-10 for more information.

Chapter 2

Installing the Texture Compression Tool

This section describes how to install the Texture Compression Tool. It contains the following sections.

- *Installing the Texture Compression Tool on Microsoft Windows* on page 2-2
- *Installing the Texture Compression Tool on Linux* on page 2-3.

2.1 Installing the Texture Compression Tool on Microsoft Windows

This section describes how to install the Texture Compression Tool on Microsoft Windows. It contains the following sections:

- *Installation requirements*
- *Procedure for installing the Texture Compression Tool on Microsoft Windows.*

2.1.1 Installation requirements

To install the Texture Compression Tool on Microsoft Windows, you require:

- Microsoft Windows XP Professional Version 2002, service pack2
- 10 MB free hard disk space.

———— **Note** —————

The Texture Compression Tool has been tested successfully on a 32-bit computer.

2.1.2 Procedure for installing the Texture Compression Tool on Microsoft Windows

To install the Texture Compression Tool on Microsoft Windows:

1. Go to the Mali Developer Center website at:
<http://www.malideveloper.com>
2. Download the Texture Compression Tool package.
3. Run the file `Mali_GPU_Texture_Compression_Tool_WinXP_vm.n.exe`.
where:
m identifies the major version
n identifies the minor version.
4. Select the required installation options and then click **Finish** to complete the installation.

By default, the Texture Compression Tool is installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Texture Compression Tool vm.n`

The Texture Compression Tool examples are installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Texture Compression Tool vm.n\examples`

2.2 Installing the Texture Compression Tool on Linux

This section describes how to install the Texture Compression Tool on Linux. It contains the following sections:

- *Installation requirements*
- *Procedure for installing the Texture Compression Tool on Linux.*

2.2.1 Installation requirements

To install the Texture Compression Tool on Linux, you require:

- Red Hat Enterprise Linux 4
- GNU tar version 1.13 or higher
- 10 MB free hard disk space.

———— **Note** —————

The Texture Compression Tool has been tested successfully on a 32-bit computer.

2.2.2 Procedure for installing the Texture Compression Tool on Linux

To install the Texture Compression Tool on Linux:

1. Locate the Mali Developer Center website at:
<http://www.malideveloper.com>
2. Download the following package:
`Mali_GPU_Texture_Compression_Tool_RHEL4_vm.n.tar.gz`
where:
m identifies the major version
n identifies the minor version.
3. To decompress the file:
 - open a command terminal and navigate to the directory where you have downloaded the package
 - type the following command:
`tar -zxvf Mali_GPU_Texture_Compression_Tool_RHEL4_vm.n.tar.gz`

By default, the Texture Compression Tool is installed in:

`ARM/Mali_Developer_Tools/Mali_GPU_Texture_Compression_Tool_vm.n`

The Texture Compression Tool examples are installed in:

ARM/Mali_Developer_Tools/Mali_GPU_Texture_Compression_Tool_vm.n/examples/

Chapter 3

Using the Texture Compression Tool

This chapter describes how to use the Texture Compression Tool. It contains the following sections:

- *Starting the Texture Compression Tool* on page 3-2
- *The Texture Compression Tool GUI* on page 3-3
- *Setting the output directory* on page 3-6
- *Opening textures* on page 3-5
- *Compressing textures* on page 3-7
- *Using the Texture Compression Tool on the command line* on page 3-10.

3.1 Starting the Texture Compression Tool

This section describes the basic operation of the Texture Compression Tool. It contains the following sections:

- *Starting the Texture Compression Tool from Microsoft Windows*
- *Starting the Texture Compression Tool from Linux.*

3.1.1 Starting the Texture Compression Tool from Microsoft Windows

To start the Texture Compression Tool from Microsoft Windows, select:

Start → **All Programs** → **ARM** → **Mali Developer Tools** → **Mali Texture Compression Tool**

3.1.2 Starting the Texture Compression Tool from Linux

To start Texture Compression Tool from Linux:

1. Open a command terminal
2. Navigate to:
`ARM/Mali_Developer_Tools/Mali_Texture_Compression_Tool_vm.n/bin`
where:
m identifies the major version
n identifies the minor version.
3. Set the LD_LIBRARY_PATH to define the location of the QT shared object files.
If you are using a Bourne shell `sh`, use the command:
`export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:`pwd``
If you are using a C shell `csh`, use the command:
`setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:`pwd``
4. Type the following command to run the executable:
`./tctools`

3.2 The Texture Compression Tool GUI

When the Texture Compression Tool is opened it displays a window with two menus, six buttons and two empty panes.

When you open one or more textures a preview of each texture is displayed in the left pane. You can then click on a preview to view a texture in the right pane.

Figure 3-1 shows the Texture Compression Tool with textures loaded.

When the cursor is within the main pane, the selected texture, displayed in the main pane, can be scaled to larger and smaller sizes by using the scroll wheel in your mouse, or using the + and - keys on your keyboard. To display the texture at its original size, right click on a zoomed texture at any time and select **Original size**.

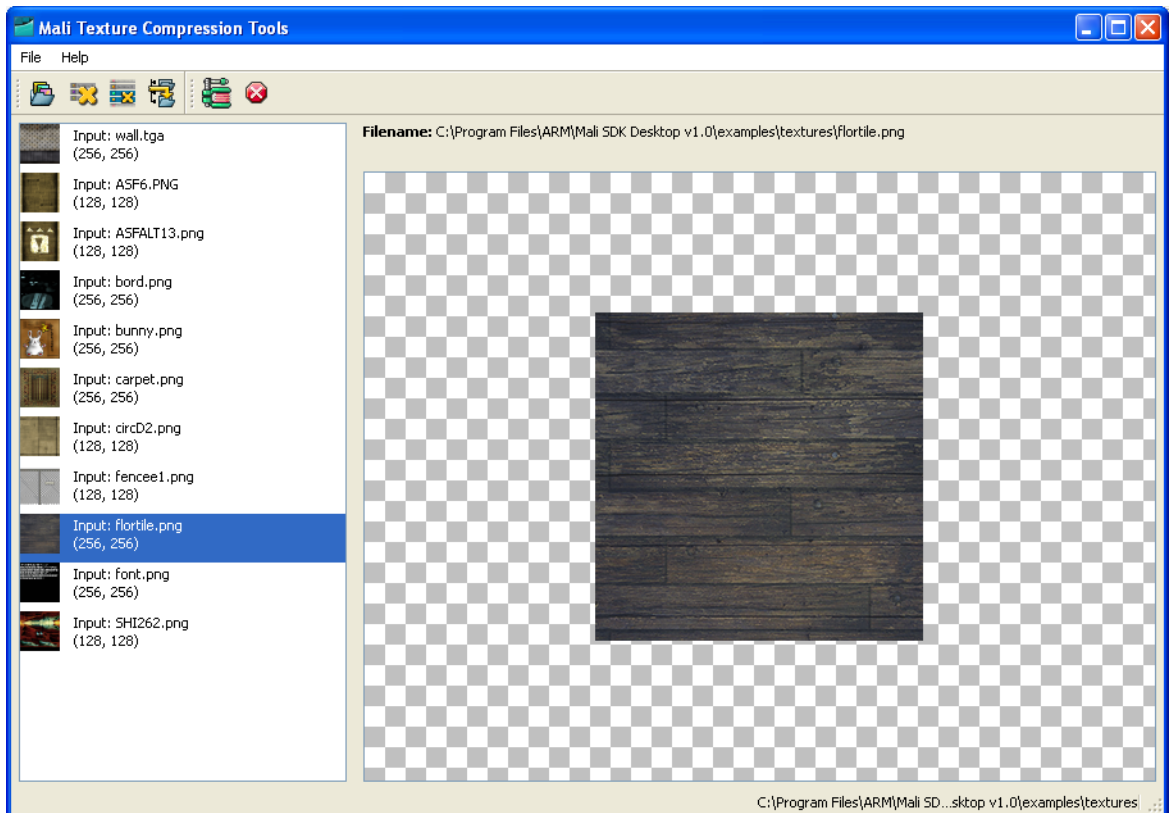








Figure 3-1 Texture Compression Tool window

You can control the Texture Compression Tool using menu options, key strokes or using the buttons on the GUI. Table 3-1 shows the Texture Compression Tool buttons.

Table 3-1 Texture Compression Tool GUI buttons

Button	Use this button to...
	Open a texture for compression.
	Remove all textures from the GUI tool window.
	Remove all selected textures from the GUI tool window.
	Set the output directory where compressed textures are saved.
	Compress a texture. Clicking this button displays the Compression Options window.
	Stop compression.

3.3 Opening textures

To open a texture for compression:

1. From the menu, select **File** → **Open Images**.
The **Select images to compress** window opens.
2. Navigate to the required texture and click **Open**. The texture appears in the left pane of the Texture Compression Tool.
3. Click on the texture in the left pane of the Texture Compression Tool to display it in the main pane.

3.4 Setting the output directory

To set the directory that the Texture Compression Tool saves compressed files to:

1. Do one of the following:

- Select **File** → **Set output directory**
- Press **CTRL+D**
- Click the **Set output directory** icon on the toolbar.

The **Browse for folder** window opens.

2. Navigate to the directory where you want compressed files to be saved and click **OK**.

The output directory name is displayed in the status bar at the bottom of the screen.

———— **Note** —————

If you do not specify an output directory, the default directory is the location of the first input file.

—————

3.5 Compressing textures

The Texture Compression Tool GUI enables you to compress a single texture or multiple textures.

Before compressing a texture, you must first open it. See *Opening textures* on page 3-5.

To compress a texture using the Texture Compression Tool:

1. Click a texture on the left pane of the Texture Compression Tool to select it.

———— **Note** ————

You can select multiple textures for compression by holding down the **CTRL** key while selecting images with the mouse.

2. To compress the texture, do one of the following:
 - Click the **Compress** button on the toolbar.
 - Select **Compress** from the **File** menu.
 - Press **spacebar**.

The **Compression options** dialog box displays.

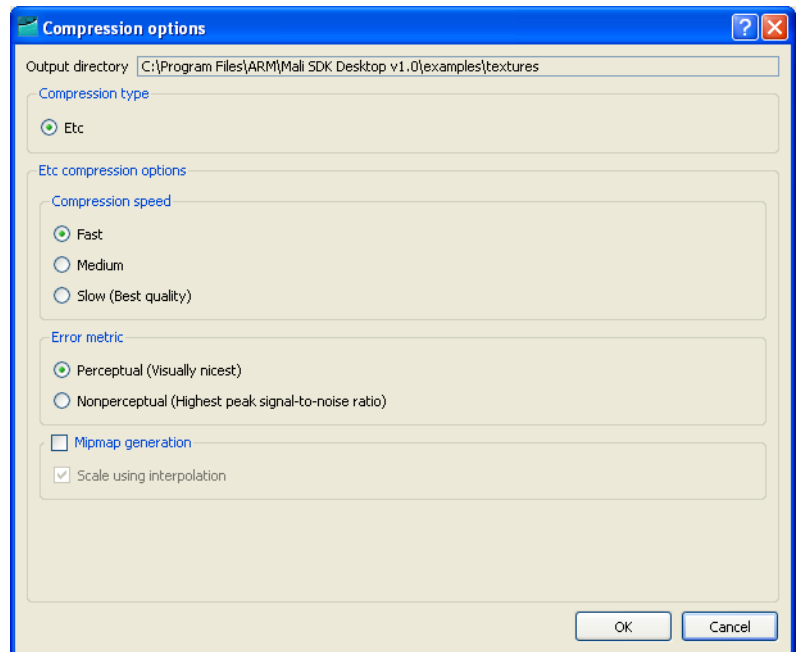


Figure 3-2 Compression options

Note

If you are compressing a single texture, you can right-click on it in the left pane and select **Compress selection** as an alternative to steps 1 and 2.

3. Select the compression options described in Table 3-2.

Table 3-2 Options for ETC compression

Option	Description
Compression speed	Select a compression speed. Slower speeds provides better image quality.
Error metric	<p>The options are:</p> <ul style="list-style-type: none"> • perceptual provides the best visual results. Using perceptual, the compression algorithm sets green closer to its required value, at the expense of an inferior representation of red and blue. This decreases the <i>Peak Signal-to-Noise Ratio</i> (PSNR), but provides a superior visual result because the eye is more sensitive to green than to blue and red. • nonperceptual is optimized to provide the highest PSNR. This is the default. Although technically superior, this setting does not account for the fact that the eye is more sensitive to green than to blue and red, and images might appear to be visually inferior.
Mipmap generation	Check this option to generate mipmaps. Using the Scale using interpolation option causes the application to blend several pixels to produce a given pixel of the downscaled textures. This gives a smooth result. The alternative method, scaling using the nearest pixel, produces a less smooth result, but might be preferable for textures or images that contain lots of text.

4. Click **OK**. The texture is compressed and the Texture Compression Tool displays the results. Figure 3-3 on page 3-9 shows an example results window.

The compression results window shows the following:

- The names of the input and output files.
- Original texture and compressed texture so you can compare the two textures.

- A view showing the difference between the RGBA values of the original and compressed textures. With good quality compression, there is little difference between the uncompressed and compressed versions, resulting in a low RGBA difference. In such cases, you can move the **Brightness factor** slider to the right to obtain an improved view of the difference. Place the cursor over any part of a texture to view RGBA values for the original and compressed versions.
- A summary of the compression options used in the compression.
- If you selected mipmap generation, a slider that enables you to vary the displayed mipmap level. The lowest mipmap level is zero, and uses the original texture. Setting successive higher levels enables you to use the next available mipmap in the series, which is half the size.

The selection in the left pane now shows a green bar to indicate that the texture has already been compressed.

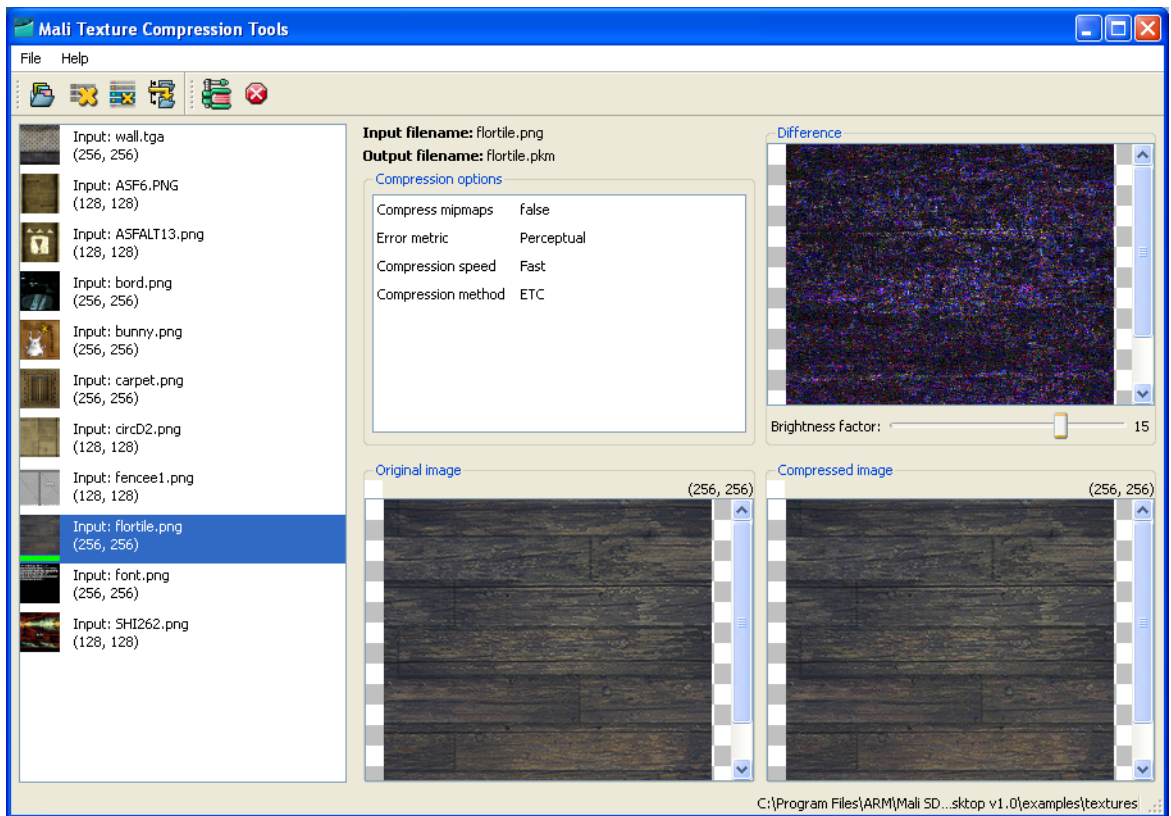


Figure 3-3 Compression results

3.6 Using the Texture Compression Tool on the command line

This section describes how to use the Texture Compression Tool from the command line. It contains the following sections:

- *About using the Texture Compression Tool on the command line*
- *Starting the Texture Compression Tool from the command line on Microsoft Windows*
- *Starting the Texture Compression Tool from the command line on Linux on page 3-11*
- *Compressing files from the command line on page 3-11.*

3.6.1 About using the Texture Compression Tool on the command line

You can use the Texture Compression Tool from the command line, this enables you to compress textures using a single command. You can use arguments and options to specify:

- texture compression algorithm to use
- name of the file to compress
- output directory to store compressed file
- texture compression options specific to the compression algorithm used.

3.6.2 Starting the Texture Compression Tool from the command line on Microsoft Windows

To start the Texture Compression Tool on Microsoft Windows:

1. Select **Run...** from the **Start** menu.
2. Type `cmd` and click **OK** to open a command line.
3. Navigate to the following directory:

Program Files\ARM\Mali Developer Tools\Mali Texture Compression Tool *vm.n*\bin
where:

m identifies the major version
n identifies the minor version.

4. Type the following command
`malitc`

The `malitc` command without arguments or options displays help information about the command.

3.6.3 Starting the Texture Compression Tool from the command line on Linux

To start the Texture Compression Tool Linux:

1. Open a command line.
2. Navigate to the following directory:
ARM/Mali_Developer_Tools/Mali_Texture_Compression_Tool_vm.n/bin
where:
m identifies the major version
n identifies the minor version.
3. Set the LD_LIBRARY_PATH to define the location of the QT shared object files.
If you are using a Bourne shell sh, use the command:
export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:`pwd`
If you are using a C shell csh, use the command:
setenv LD_LIBRARY_PATH \${LD_LIBRARY_PATH}:`pwd`
4. Type the following command
./malitc

The malitc command without arguments or options displays help information about the command.

3.6.4 Compressing files from the command line

Use the following command to compress a texture:

```
malitc [input file] [output directory] <options>
```

The arguments are:

[input file] The input file to be compressed.

[output directory]

The directory to place the compressed file into.

You must specify an output directory. If an output directory is not specified, the Texture Compression Tool requests one.

The options are:

```
-s [fast|medium|slow]
```

Compression speed:

- slow provides best image quality

- fast is the default.

-e [perceptual|nonperceptual]

See *Options for ETC compression* on page 3-8.

-mipmaps Specify this option to generate mipmaps. A mipmap is a collection of scaled-down bitmap images that accompanies a main texture. Using mipmaps can increase rendering speed and reduce artifacts.

Glossary

This glossary describes the terms used in *Media Processing Division* (MPD) documents from ARM Limited.

AEL *See* ARM Embedded Linux.

ARM Embedded Linux (AEL)

A version of embedded Linux OS ported to the ARM architecture.

DCC *See* Digital Content Creation.

DDK *See* Mali DDK.

Demo Engine *See* Mali Demo Engine.

Digital Content Creation (DCC)

The creation and modification of custom content such as animations, graphics, textures, and images, before presentation in the final medium.

EGL *See* Embedded-System Graphics Library.

EGL driver *See* Native platform graphics interface.

Embedded-System Graphics Library (EGL)

A standardized set of functions that communicate between graphics software, such as OpenGL ES or OpenVG drivers, and the platform-specific windowing system that displays the image.

- ESSL** *See* OpenGL ES Shading Language.
- ESSL compiler** The compiler that translates shaders written in ESSL, into binary code for the shader units in the Mali GPU. There are two versions of the ESSL compiler:
- the on-target compiler
 - the offline compiler.
- Fixed-function pipeline** A process that uses standard functions to draw graphics on fixed-function graphics hardware, such as the Mali-55 GPU. The fixed-function pipeline is a requirement of OpenGL ES 1.1.
- Fragment** A fragment consists of all data, such as depth, stencil, texture, and color information, required to generate a pixel in the framebuffer. A pixel is usually composed of several fragments. A fragment can be multi-sampled and super-sampled.
- Fragment processor** A programmable component of the Mali GPU pixel processor, that runs fragment shaders.
- Fragment shader** A program running on the Mali GPU pixel processor that calculates the color and other characteristics of each fragment.
- Geometry processor** A geometry processor is a component of a programmable-function Mali GPU. It executes vertex shaders with typical transform and lightning calculations, and generates lists of primitives for a pixel processor to draw.
- GPU** *See* Graphics Processor Unit.
- Graphics application** A custom program that executes in the Mali graphics system and displays graphics content in a framebuffer for transfer to a display.
- Graphics driver** A software library implementing OpenGL ES or OpenVG, using graphics accelerator hardware.
- See also* OpenGL ES driver and OpenVG driver.
- Graphics Processor Unit (GPU)** A hardware accelerator for graphics systems using OpenGL ES and OpenVG. The hardware accelerator comprises of an optional geometry processor and a pixel processor together with memory management. Mali programmable GPUs, such as the Mali-200 and Mali-400 MP GPUs, consist of a geometry processor and at least one pixel processor. Mali fixed-function GPUs, such as the Mali-55 GPU consist of a pixel processor only.
- See also* Geometry Processor, Pixel Processor, Mali MMU.

Instrumented drivers

Alternative graphics drivers that are used with the Mali GPU. The Instrumented drivers include additional functionality such as error logging and recording performance data files for use by the Performance Analysis Tool.

See also Mali DDK, Performance Analysis Tool.

Mali

The name given to graphics software and hardware products developed by ARM, that accelerates 2D and 3D graphic applications.

Mali Binary Asset

A proprietary file format that you can load onto the Mali Demo Engine and use on Mali GPUs. Mali Binary Asset is faster to parse, uses less memory, and is smaller in size than formats such as COLLADA XML. If a deployment format is not available, use the Mali GPU Binary Asset Exporter to convert graphics assets into Mali Binary Asset.

Mali DDK

A set of drivers, typically for AEL, that enable communication between AEL and the Mali GPU. These drivers are available as normal or instrumented versions.

See also Instrumented drivers.

Mali MMU

A full-featured *Memory Management Unit* (MMU) that is present on Mali GPUs such as the Mali-200 GPU and the Mali-400 MP GPU.

Mali Remote Interface

The Mali Remote Interface is a component of the Instrumented driver. This interface enables the Performance Analysis Tool to access performance data dump files over a network interface.

See also Instrumented drivers, Performance data file.

Mali Surface

The destination for Mali GPU output. It can potentially be for color, depth and stencil, however, when referring to EGL surfaces, it only refers to color output buffers.

Mali Demo Engine Library

A C++ class framework for developing OpenGL ES 2.0 applications for the Mali GPU.

Multi-sampling

An anti-aliasing technique where each pixel in the framebuffer is split into multiple samples corresponding to different positions within the area covered by the pixel. The Mali GPU pixel processors support multi-sampling at four samples per pixel with negligible performance impact.

Native platform graphics interface (EGL) driver

A standardized set of functions that communicate between graphics software, such as OpenGL ES or OpenVG drivers, and the platform-specific window system that displays the image.

- Offline Compiler** A command line tool that translates vertex shaders and fragment shaders written in the ESSL into binary vertex shaders and binary fragment shaders that you can link and run on the Mali GPU.
- On-target compiler** A component of the Mali GPU OpenGL ES 2.0 driver that translates shader source files provided by the graphics application, into binary shader code, at runtime.
- OpenGL ES driver** The OpenGL ES driver is part of a driver stack that translates OpenGL ES API commands into data and instructions for the Mali GPU. Only the device driver controls the Mali GPU directly.
- OpenGL ES Shading Language (ESSL)**
A programming language used to create custom shader programs that can be used within a programmable pipeline, on the Mali GPU. You can also use pre-defined library shaders, written in ESSL.
- OpenVG driver** The OpenVG driver is part of a driver stack that translates OpenVG API commands into data and instructions for the Mali GPU. Only the device driver controls the Mali GPU directly.
- Performance Analysis Tool**
A fully-customizable GUI tool that displays and analyzes performance data files produced by the Instrumented drivers, together with framebuffer information.

See also Instrumented drivers, Performance data file.
- Render Target** *See* Mali Surface
- Performance counter**
Data produced by the Instrumented drivers and the Mali GPU, that can be displayed and analyzed as statistical information in the Performance Analysis Tool.
- Performance data file**
Files that contain a description of the performance counters, together with the performance counter data in the form of a series of values and images. Performance data files are saved in .ds2 format and can be loaded directly into the Performance Analysis Tool.
- Performance variable**
Data produced by the Instrumented drivers and the Mali GPU, that can be displayed and analyzed as statistical information in the Performance Analysis Tool.
- Pixel processor** A pixel processor, such as the Mali-200 GPU pixel processor, performs rendering operations to produce a final image for display. The Mali-200 and Mali-400 MP GPU pixel processors receive completed vertex data from the Mali GPU geometry processor.

Primitive	<p>A basic element that is used, by the Mali GPU, with other primitives, to generate images. A primitive can be a point, a line, a triangle, or a quad. Properties of primitives are defined through the use of vertices. Each primitive is divided into fragments so that there is one fragment for each pixel covered by the primitive.</p> <p><i>See also</i> Vertex.</p>
Programmable pipeline	<p>A process that uses custom programs to draw graphics on programmable graphics hardware, such as the Mali-200 and Mali-400 MP GPUs. The programmable pipeline is a requirement of OpenGL ES 2.0.</p>
Rasterization	<p>The process within the Mali GPU pixel processor that identifies the fragment of each triangle that is seen through each pixel on the display screen.</p>
Rasterizer	<p>A unit or method to convert a geometrical description of primitives supported by the Mali GPU: point, line, triangle, or quad, to fragments. The rasterizer works on line equations generated in the triangle setup phase of Mali GPU pixel processors.</p>
Render	<p>Rendering in the context of the Shader Debugger plug-in refers to the sending of shader effects to a remote Renderer, so that a preview image can be generated.</p>
Renderer	<p>A device capable of rendering shader effects and returning an image of the rendered shader. Typically, this is a device with a Mali GPU, though it can also be a local or remote emulation.</p>
Render Target	<p><i>See</i> Mali Surface.</p>
Shader	<p>A Shader or Shader Source file, refers to a single source file for a shader effect. This source file might be a vertex shader, a fragment shader or a non-specific utility shader that can be linked concatenated and compiled with other shaders.</p> <p><i>See also</i> Fragment shader, Vertex shader</p>
Texture Compression Tool	<p>A component of the Mali Developer Tools that you can use to compress textures and images, using the ETC algorithm.</p>
Vertex	<p>A set of data defining the properties of one point of a primitive. For example, a point primitive, an endpoint of a line primitive, or a corner of a triangle primitive.</p>
Vertex shader	<p>Program vertex shader data sent to the Mali GPU geometry processor. The geometry processor calculates the position and other characteristics, such as color and texture coordinates, for each vertex.</p>
Vertex shader unit	<p>A programmable component of the Mali GPU geometry processor, that runs vertex shaders.</p>

Vertices

This is the plural form of the word vertex.

See also Vertex.