

N I N T E N D O  
**NITRO**-System

G2D Library

Release Notes

Version 1.2.5

**The contents in this document are highly  
confidential and should be handled accordingly.**

**Confidential**

These coded instructions, statements, and computer programs contain proprietary information of Nintendo of America Inc. and/or Nintendo Company Ltd. and are protected by Federal copyright law. They may not be disclosed to third parties or copied or duplicated in any form, in whole or in part, without the prior written consent of Nintendo.

# Table of Contents

1	G2D Library.....	8
1.1	Runtime Library.....	8
1.2	Multithread Operations.....	8
1.3	Binary Converter.....	8
2	Major Revisions .....	9
2.1	Changes from Version 1.2.4 .....	9
2.1.1	General.....	9
2.1.2	OAM Manager .....	9
2.1.3	BG .....	9
2.1.4	Drawing Characters.....	9
2.1.5	Converter.....	9
2.1.6	fontcvtr.....	10
2.2	Changes from Version 1.2.3 .....	10
2.2.1	General.....	10
2.2.2	Converter.....	11
2.2.3	Cell Animations.....	11
2.2.4	Multi-cell Animations.....	11
2.2.5	BG .....	11
2.3	Changes from Version 1.2.2 .....	11
2.3.1	General.....	11
2.3.2	Data Structures.....	12
2.3.3	Converter.....	12
2.3.4	Animation Controller.....	12
2.3.5	Cell Animations.....	12
2.3.6	Multicell Animations.....	12
2.3.7	OAM Manager .....	13
2.3.8	Extended OAM Manager.....	13
2.3.9	Renderer.....	13
2.3.10	BG .....	13
2.3.11	Drawing Text.....	13
2.4	Changes from Version 1.2.1 .....	14
2.4.1	General.....	14
2.4.2	Cell Animations.....	14
2.4.3	Drawing OAM Software Sprites.....	14
2.4.4	Multicell Animations .....	14
2.4.5	Renderer and Renderer Core .....	14
2.4.6	Converters .....	15

2.5	Changes from Version 1.2.0 .....	15
2.5.1	General .....	15
2.5.2	Multicell Animation .....	15
2.5.3	Extended OAM Manager .....	15
2.5.4	Renderer .....	15
2.5.5	Converter .....	15
2.5.6	Other .....	15
2.6	Changes from Version 1.1.0 .....	16
2.6.1	Data Structures .....	16
2.6.2	Multicell Animation .....	16
2.6.3	OAM Software Sprite Rendering.....	16
2.6.4	Converter .....	16
2.7	Changes from Version 1.0.0 .....	17
2.7.1	General .....	17
2.7.2	Multicell Animation .....	17
2.7.3	Extended OAM manager .....	17
2.7.4	Software Sprite.....	17
2.7.5	Graphic Image Load Related .....	17
2.7.6	Converter .....	17
2.8	Changes from Version 0.9.5 .....	18
2.8.1	General .....	18
2.8.2	Cell.....	18
2.8.3	Software Sprite.....	18
2.8.4	OAM Software Sprite Rendering.....	18
2.8.5	Changes Related to Graphical Image Load .....	18
2.8.6	Renderer .....	19
2.8.7	Converter .....	19
2.9	Changes from Version 0.9.0 .....	19
2.9.1	General .....	19
2.9.2	Cells .....	19
2.9.3	Software Sprites.....	19
2.9.4	Changes Related to Graphical Image Loading.....	20
2.9.5	Renderer .....	20
2.9.6	Converter .....	20
2.10	Changes from Version 0.8.0 .....	21
2.10.1	Overall.....	21
2.10.2	OAM Manager.....	21
2.10.3	Extended OAM Manager .....	21
2.10.4	Renderer .....	21
2.10.5	Converter .....	21

2.11	Changes from Version 0.7.0 .....	22
2.11.1	OAM Manager .....	22
2.11.2	Renderer.....	22
2.11.3	Cell Animation .....	22
2.11.4	Multicell Animation.....	22
2.11.5	Converter .....	22
2.12	Changes from Version 0.6.0 .....	23
2.12.1	Animation .....	23
2.12.2	Converters .....	23
2.13	Changes from Version 0.5.0 .....	24
2.13.1	General.....	24
2.13.2	Loading Files .....	26
2.13.3	OAM Manager .....	26
2.13.4	Software Sprites .....	26
2.14	Changes from Version 0.4.0 .....	27
2.14.1	General.....	27
2.14.2	Cell Animation .....	27
2.14.3	Multicell.....	27
2.14.4	OAM Manager .....	27
2.14.5	Converter .....	27
2.15	Changes from Version 0.3.0 .....	28
2.15.1	General.....	28
2.15.2	Animation Controller.....	28
2.15.3	Extended OAM Manager.....	28
2.15.4	Renderer.....	28
2.16	Changes from Version 0.2.0 .....	29
2.16.1	File Loading .....	29
2.16.2	OAM Manager .....	29
2.16.3	Software Sprite .....	29
2.16.4	General.....	29
2.17	Changes from Version 0.1.0 .....	29
2.17.1	File Loading .....	29
2.17.2	Animation Controller.....	29
2.17.3	OAM Manager .....	30
2.17.4	Extended OAM Manager 2.....	30
2.17.5	Changes Related to Graphical Image Load .....	30
2.17.6	Entity.....	30
2.17.7	Renderer.....	30
2.17.8	Sample .....	30
2.17.9	General.....	30

3	Known Problems .....	31
4	Future Plans .....	31

## Revision History

Version	Revision Date	Description
1.2.5	2007/03/14	Added support for the March 14, 2007 version.
1.2.4	2006/05/29	Added support for the May 29, 2006 version.
1.2.3	2005/09/01	Added support for the September 1, 2005 version.
1.2.2	2005/06/06	Added support for the June 6, 2005 version.
1.2.1	2005/03/28	Added support for the March 28, 2005 version.
1.2.0	2005/01/31	Added support for the Jan 6, 2005 version.
1.0	2004/12/06	Added support for the December 6, 2004 version.
1.0.0	2004/11/18	Added support for the November 18, 2004 version.
0.9.5	2004/10/12	Added support for the October 12, 2004 version. Added new items to schedule. Added a warning about multithreaded operations.
0.9.0	2004/09/16	Added support for the September 16, 2004 version.
0.8.0	2004/09/02	Added support for the September 2, 2004 version.
0.7.0	2004/08/10	Added support for the August 10, 2004 version.
0.6.0	2004/08/02	Added support for the August 2, 2004 version.
0.5.0	2004/07/20	Added support for the July 20, 2004 version.
0.4.0	2004/06/22	Made to support the June 22, 2004 version.
0.3.0	2004/06/10	Made to support the June 10, 2004 version.
0.2.0	2004/05/28	Made to support the May 28, 2004 version.
0.1.0	2004/05/10	Initial Version.

# 1 G2D Library

## 1.1 Runtime Library

---

The NITRO-System G2D library offers features that allow high-quality 2D display using both the Nintendo DS 2D graphics engine and the Nintendo DS 3D graphics engine. The G2D library can also use the animation data that are output from the NITRO-CHARACTER 2D tool, which is supplied with NITRO-System to pattern animate characters.

For details on the G2D library, see the *G2D Library 2D Graphics Overview* (NitroSystem\docs\G2D\G2D\_Overview.pdf) or the *Function Reference*.

## 1.2 Multithread Operations

---

NITRO-System library was not designed to be fundamentally thread-safe (that is, to support multithreading).

**Note:** When called from interrupt handlers and different threads, the G2D library functions may not always work correctly.

## 1.3 Binary Converter

---

Conversion of data with the binary converter is required, because the data interpreted by G2D at runtime is created from the binary data exported from NITRO-CHARACTER.

For details about `g2dcvtr.exe`, which is used in data conversion, see *g2dcvtr Manual* (NitroSystem\docs\G2D\g2dcvtr\_Manual.pdf).



## 2 Major Revisions

### 2.1 Changes from Version 1.2.4

---

#### 2.1.1 General

---

In the library header files that used static inline, function declarations were changed from static inline to inline. A decrease in the code size can be expected due to this change.

#### 2.1.2 OAM Manager

---

Added the `NNS_G2dGetOamBuffer()` function, which obtains a pointer to the OAM Manager module's internal buffer.

#### 2.1.3 BG

---

Revised such that the region transferred in `NNS_G2dBGLoadScreenRect` will be clipped if it exceeds the range of the source or destination of the transfer.

#### 2.1.4 Drawing Characters

---

- Added support for vertical writing and vertical style display.
- To the sample demos, added the PortraitHW, PortraitSW, and DrawVertical demos.
- The OBJ1D and the MinimumCanvas sample demos were changed so that the VRAM mode and the color mode could be switched. If CharCanvas is of an illegal size, a warning or correction occurs.

#### 2.1.5 Converter

---

- In the data for which the number of displayed frames was zero in all animation frames, a specific bug caused illegal shutdown codes not to return correctly. This bug has been corrected.
- When animation frames with zero displayed frames were specified at the end of a multicell animation sequence, the end frame was being displayed mistakenly. This bug has been fixed.
- If a zero was specified for the number of displayed frames at the end frame of a cell animation sequence within a multicell animation, animations were not updated correctly in some cases due to the update timing of the multicell animation. This bug has been fixed.
- Implemented a feature that checks for illegal character numbers of OBJs within the cell. Made the illegal data cancel the conversion.
- Added the `-cza` option. This option checks the animation frames for which zero is specified as the number of the displayed animation frame, and treats these frames as illegal data.

## 2.1.6 fontcvtr

---

- Added a feature for outputting NITRO fonts for vertical writing.
- Added a feature for outputting NITRO fonts for vertical style.
- Added an option for making the font size interpretation in the same way as is generally done by Windows software, if a Windows font is being used as the input.
- Added a soft antialiasing feature. If a Windows font is input and converted to a font that is not monochrome, this feature can improve the output quality (which also depends on the input font).
- Due to the addition of the feature to output NITRO fonts for vertical writing and vertical-style, the glyph rotation feature during BMP output was removed from the GUI. It can be used with the CUI version.
- Fixed the bug that caused fonts not to be output correctly if “Input” was specified as a Windows font and “2” was set to “Levels of Gray.”
- Fixed a bug in the command line (CUI) version in which the `-iu` command line option had become `-iv`. `-iu` is the correct option, but `-iv` can also be used to maintain compatibility. (This does not appear in the manual.)
- Fixed the bug that caused the color of the width line to be processed in the same way as the color within the cell.
- Fixed the bug that caused the message window to close even when the conversion failed in the GUI version.
- Characters that have the same character codes appeared in the European character region and in the Japanese character region of the character order file (`ds_ipl.xlor`) that outputs the IPL font table. This has been corrected. The corresponding parts of the Japanese character region have been replaced by `<null/>`.
- Added a character order file to output fonts for the Chinese and Korean IPL.
- Added a character order file to output characters used by UHC (Code page 949).

## 2.2 Changes from Version 1.2.3

---

### 2.2.1 General

---

Fixed the problem of the G2D Library not operating correctly when `GX_SetDefaultDMA()` was set with `GX_DMA_NOT_USE`.

---

### 2.2.2 Converter

---

- Fixed the problem with the cell's region information calculations being output incorrectly when the cell included OBJ for which double-angle affine mode was set. Now the same region information gets output similarly to cells that only include normal OBJs.
- In version 1.2.2, the process for calculating VRAM transfer information was changed. This resulted in a problem with incorrect calculation of that information. The problem was fixed.
- Fixed the problem of incorrectly calculated cell border radius.
- Added the `-afs` option, which aligns the size of the output file to be a multiple of 4 bytes.

---

### 2.2.3 Cell Animations

---

Added the `NNS_G2dRestartCellAnimation()` function, which restarts animation playback from the beginning of the animation.

---

### 2.2.4 Multi-cell Animations

---

- Added the `NNS_G2dStartMCCellAnimationAll()` function, which sets the cell animations that comprise the multicell to the 'start animation play' state.
- Added the `NNS_G2dRestartMCAnimation()` function, which restarts playback of the multicell animation from the beginning of the animation.

---

### 2.2.5 BG

---

Fixed the bug which prevented the correct loading of the screens whose width and height were not multiples of 32 characters. With this, changed the arguments of the `NNS_G2dBGLoadScreenRect` function.

---

## 2.3 Changes from Version 1.2.2

---

---

### 2.3.1 General

---

- Extended the supported file formats to include the `NCGR` and `NCBR` character data file formats, adding a character position data block. Also added the function `NNS_G2dGetUnpackedCharacterPosInfo()` to access this data block.
- Added the sample demo `Renderer_CharChange`. This demo switches among multiple images for the renderer. It also shows how to use the partial character output option `-br`.
- Added the sample demo `UserExAttribute`. This demo uses the attributes extended by the user.
- Added the sample demo `MultiCell_UILayout`. This demo performs user interface processing for game applications that use multicells.

---

### 2.3.2 Data Structures

---

- Added an accessor API to enable the use of the attributes extended by the user.
- Added `NNS_G2dGetAnimSequenceIndex()`.

---

### 2.3.3 Converter

---

- Added the option `-br/` to enable the conversion processing to be performed by specifying a rectangular portion of the character file.
- Corrected a problem where a transfer size larger than necessary was specified when VRAM transfer data was output.
- Corrected a problem where an invalid warning message was output during normal conversion processing. The warning displays information about an invalid type conversion that causes truncation when converting variable types within the converter. In previous versions, even valid type conversions outputted this warning.
- Corrected a problem where the screens with a height other than 256X were not being converted properly.
- Corrected the problem where the rectangular region size was output one size larger than the actual region size.
- Added the `-rtp` option so that transparent pixels would get removed from the border area before calculating border area data.
- Added the `-oua` option to extract comment strings in extended comment fields that are added in the `ncc` file v1.04 as extended user attribute data. (For details, see the *g2dcvtr\_Manual* manual.)

---

### 2.3.4 Animation Controller

---

Added new functions, `NNS_G2dGetAnimCtrlCurrentAnimSequence()` and `NNS_G2dGetAnimCtrlCurrentElemIdxVal()`.

---

### 2.3.5 Cell Animations

---

Corrected the failed rendering of animation sequences that specified a zero display time for the leading animation.

---

### 2.3.6 Multicell Animations

---

- Corrected a problem where animation update was not performed correctly for the data whose multicell Node attribute and `NNS_G2D_MCANIM_PLAYMODE_CONTINUE` had been set.

**Note:** Due to the correction of this bug, the computational load related to updating multicell animations increased to some degree. Specifically, the average execution time of `NNS_G2dTickMCAnimation()` increased by 19% (from 57  $\mu$  seconds to 68  $\mu$  seconds) in the `Renderer_PerfCheck` sample. The degree to which the computational load increases depends on the data.

The following are the characteristics of additional processing:

- Processing occurs during multicell switching.
  - If `NNS_G2D_MCANIM_PLAYMODE_CONTINUE` is not set, processing will not be performed.
  - As the total number of animation frames accessed by multicell for cell animation increases, so does the processing load.
- Added functions `NNS_G2dTraverseMCCellAnims()` and `NNS_G2dTraverseMCNodes()`.

---

### 2.3.7 OAM Manager

Corrected a problem where assert would fail if the transformation enable flag was not set for the OBJ, and the OBJ was registered by the OAM manager with an affine reference number. (Operations were changed so that processing would continue without setting the affine number.)

---

### 2.3.8 Extended OAM Manager

Corrected a problem where an invalid assert warning would be displayed if NULL was set for `NNSG2dOamExEntryFunctions.getAffineCapacity` and `NNSG2dOamExEntryFunctions.funcs.entryNewAffine`. This would occur even if the affine parameters were not used when the registration functions for the extended OAM manager were set using `NNS_G2dSetOamManExEntryFunctions()`.

---

### 2.3.9 Renderer

Corrected a problem in which cells were not drawn correctly. This occurred if OBJ rendering was done on a cell that did not use affine transformation after a software sprite was drawn for a cell that did use affine transformation.

---

### 2.3.10 BG

- Corrected a problem in `NNS_G2dBGSetup` and `NNS_G2dBGLoadElements`. The extended palette was being loaded into the wrong slot with the sub-screen as the target.
- A problem was corrected in `NNS_G2dBGSetup`, where the wrong BG mode would result if an attempt was made to read an affine-expanded BG into BG3 when BG3 was to be an affine-expanded BG.
- Added support for loading compressed palettes and partial characters.
- Added support for loading free-size screens.

---

### 2.3.11 Drawing Text

- Corrected a problem in `NNS_G2dTextCanvasDrawTaggedText`, in which changes made using a tag process callback to `TextCanvas` were not reflected in the drawing itself.
- Added a function for creating cells that display `CharCanvas` using the renderer.

---

## 2.4 Changes from Version 1.2.1

---

### 2.4.1 General

---

- Fixed the problem where cells with the OBJs positioned outside of the -128 and 127 range were drawn incorrectly. The OBJ position inside of a cell can be set to a value between -256 and 255. You can revert to the previous drawing functionality by defining `NNS_G2D_LIMIT_CELL_X_128` in the `include/fmt/g2d_Cell_data.h` file, and recompiling the library.
- Reversed the direction of flat surfaces on square polygons in software sprites. As a result of this change, sprites will display correctly even when the culling mode is set to `GX_CULL_BACK`.
- Added functions for font manipulation and text character / text string drawings.
- Added a font converter (`fontcvtr.exe`) that creates font resources used in the aforementioned functions. For instructions on use, see the *g2dcvtr Manual* (NitroSystem\docs\G2D\fontcvtr\_Manual.pdf).

### 2.4.2 Cell Animations

---

- When an animation sequence whose animation frame with a 0 display time (the video frame count of 0) was played, an infinite loop occurred. We added an assert statement that treats as invalid the animation sequences that have only animation frames with a zero display time.
- Changed the library so that OBJ position correction is only used in `NNS_G2dMakeCellToOams()` when the double-size affine flag is enabled and when the OBJ's double-size flag is changed.

### 2.4.3 Drawing OAM Software Sprites

---

Added support for drawing OBJs with an enabled double-size affine flag.

### 2.4.4 Multicell Animations

---

Added `NNS_G2dSetMCAnimationCellAnimFrame()`, which sets the playback animation frame for each cell animation in a multicell animation.

### 2.4.5 Renderer and Renderer Core

---

- The OBJ position correction processing, used when the double-size affine flag overwrite is specified with `NNS_G2dSetRendererAffineOverwriteMode()`, will as of now be performed only when the OBJ's double-size flag is changed.
- Changed the processing method which uses the 2D graphics engine to draw the OBJs that had their double-size affine flag enabled by NITRO-CHARACTER. Assuming that the tool will perform OBJ position correction, the following will occur in the new version. When OBJs with an enabled double-size affine flag are entered,  $\text{OBJ size} * 1/2$  will be added to the OBJ position calculation, and the double-size affine OBJ correction value will be cancelled before the process is performed. (If you want to revert to the previous operation, comment out the `NNS_G2D_ASSUME_DOUBLEAFFINE_OBJPOS_ADJUSTED` definition in the `g2d_config.h` file.)

---

## 2.4.6 Converters

---

- Added the `-ncn` option, which uses `nce` file titles as output `NCGR` (`NCBR`) file titles. Use it when several 1D mapping format `nce` files are referencing a single `ncg` file.
- Corrected bugs related to `BuildNENR.exe`. (Mainly path interpretation errors.)
- Added warning to prevent the output of invalid data. This will serve the animation sequences that do not contain any animation frames, and the sequences that contain only the animation frames that have a display time of zero.

---

## 2.5 Changes from Version 1.2.0

---

---

### 2.5.1 General

---

Added a sample that uses the extended OAM manager in combination with the renderer.

---

### 2.5.2 Multicell Animation

---

A bug in the `NNS_G2dSetMCAnimationSpeed()` and `NNS_G2dResetMCCellAnimationAll()` functions caused them to malfunction under certain initialization conditions. That is, when `NNS_G2D_MCTYPE_DONOT_SHARE_CELLANIM` was specified as the actual multicell type during initialization, the function did not operate correctly.

---

### 2.5.3 Extended OAM Manager

---

- Changed the drawing order of affine-transformed OAMs so that it would match the drawing order of the ordinary OAMs. Previously, affine-transformed OAMs were drawn in the opposite order from the order in which they were saved. To return to the behavior in the previous implementation, define `NNS_G2D_OAMEX_USE_OLD_REINDEXOAMCHUNKLIST` and rebuild the library.
- Added `NNSG2d_SetOamManExDrawOrderType()`, which allows you to change the behavior related to the OBJ drawing order for the extended OAM manager.

---

### 2.5.4 Renderer

---

Fixed the bug related to the sharing of affine parameters in the renderer module.

---

### 2.5.5 Converter

---

When the object compression function was OFF and the 1D mapping mode data were used, corrupt conversion binaries were generated. This bug was fixed.

---

### 2.5.6 Other

---

Corrected statement errors in the Assert macro in the `g2d_SRTControl.h` file.

## 2.6 Changes from Version 1.1.0

---

### 2.6.1 Data Structures

---

Added the `NNS_G2dCalcAnimSequenceTotalVideoFrames()` function, which calculates the total video frame length of the animation sequence data.

### 2.6.2 Multicell Animation

---

- Added the `NNS_G2dResetMCCellAnimationAll()` function, which resets the animation frames for the cell animation that makes up the multicell entity.
- Changed the multicell animation initialization functions. Revised the sample demo to show how to use the functions. Made the usage method into a sample demo. Depending on the characteristics of the multicell animation data, an improved performance and memory efficiency can be anticipated (this is effective in the case where data references the same cell animation). The existing functions also remain.

### 2.6.3 OAM Software Sprite Rendering

---

Fixed a bug in the calculation method of the UV value used in software sprite rendering, when a 1D mapping with a 256-color character was being used.

### 2.6.4 Converter

---

- Corrected the following bug: the related file path information in the data block of the link file name failed to convert the file registered with an absolute path.
- Made changes such that the expansion comment information, in the label definition header file output with the `-lbl` option, was inserted as a C comment.
- Fixed a bug where the information about cell animation playback mode was not configured properly when converting multicell data.
- Deleted the feature for outputting files in C source code format.



---

## 2.7 Changes from Version 1.0.0

---

### 2.7.1 General

---

Updated the contents of the technical document and moved a section of the overview manual to API reference.

### 2.7.2 Multicell Animation

---

Added the `NNS_G2dGetMCBankNumNodesRequired()` function.

### 2.7.3 Extended OAM manager

---

With the lower priority OAM (drawn in the back), the number of drawing times was set to a smaller value. This problem was fixed.

### 2.7.4 Software Sprite

---

When `NNS_G2dSetSpriteAttrEnable()` was executed for the second time or beyond, the value was not updated properly. This problem was solved.

### 2.7.5 Graphic Image Load Related

---

For `NNS_G2dLoadImage*()` and similar functions, fixed the problem where the cache contents were not flashed before the DMA transfer.

### 2.7.6 Converter

---

- Added the following options to the converter `BuildNENR.exe`:
  - `-o/...` Output destination folder specification
  - `-src` Source file output specification
- Added the options `-ai`, `-aisrt`, and `-ait` to specify the method of output animation element in `g2dcvtr.exe`.

## 2.8 Changes from Version 0.9.5

---

### 2.8.1 General

---

- Added a renderer core module, which is created by separating from the renderer module only the essential parts necessary for render processing. The module was created with the assumption that users would want to substantially customize their process or speed up simple processes. The renderer module uses the renderer core module internally.
- Added six samples.

### 2.8.2 Cell

---

- Added `NNS_G2dGetCellAnimationCurrentCell()`.
- Added a process in the cell (multicell) SRT animation for the data that use only translational animation. Previously, even with translational animation being used exclusively, scale and rotation were set to cell (multicell) animation. With this addition, unnecessary affine transformation settings are avoided.

### 2.8.3 Software Sprite

---

Added a `NNS_G2dDrawSprite*Fast()` function, which does not save the current matrix of the 3D graphics engine. Because the function does not push or pop on the matrix stack, it executes at a high speed.

### 2.8.4 OAM Software Sprite Rendering

---

- Added the `NNS_G2dDrawOneOam3DDirect*Fast()` function, which does not save the current matrix of the 3D graphics engine. Because the function does not push or pop on the matrix stack, it executes at a high speed.
- Added the API to increase the efficiency of the rendering process by caching the UV parameter calculation results (`NNS_G2dSetOamSoftEmuSpriteParamCache()` and `NNS_G2dDrawOneOam3DDirectUsingParamCacheFast()`). The usable parts are limited, but the API operates about 50% faster than the normal `NNS_G2dDrawOneOam3DDirect*Fast` function.

### 2.8.5 Changes Related to Graphical Image Load

---

- Fixed the bug in `NNS_G2dLoadPaletteEx()`, where the palette did not load correctly.
- Added a process that write-returns the memory cache of the transfer source data region to the main memory, before the data is transferred to VRAM with `NNS_G2dLoadPalette()` and `NNS_G2dLoadPaletteEx()`.

---

## 2.8.6 Renderer

---

- Separated parts of the rendering process from the matrix stack management and parameter management. Made the rendering process parts into a separate renderer core module.
- Compared with the multicell rendering process (see sample demo `Renderer_PerfCheck`), the following improvements were observed in the process efficiency:
  - When optimization hint is not set, OBJ rendering was about -40%, and software sprite rendering was -30%.
  - When optimization hint was set, OBJ rendering was about -50%, and software sprite rendering was -40%.
- Removed `NNS_G2dOptimizeRenderer()` and added `NNS_G2dBeginRenderingEx()` as a substitute function.
- Added `NNS_G2dSetTrans()`.
- Added the items (such as `NNS_G2dSetRendererOverwriteEnable/Disable()`) which the renderer can specify in the parameter overwrite process performed for OAM.

---

## 2.8.7 Converter

---

In cell SRT animation, we added animation result types supporting data using only translational animation. Support for that output was added as well. With this extension, animation data capacity was reduced, and unnecessary affine transformation settings during execution were avoided.

---

## 2.9 Changes from Version 0.9.0

---

---

### 2.9.1 General

---

- The DMA channels used by the G2D library are now the same channels that are used by the GX library.
- To improve processing efficiency, some internal library functions were in-lined.

---

### 2.9.2 Cells

---

Added support for cell information that contains rectangle data for cell boundary information. Also, the status of the pre-existing functions (such as `NNS_G2dGetCellBoundingRect()`) used to obtain bounding sphere information was changed from internally released to publicly released.

---

### 2.9.3 Software Sprites

---

- Fixed the bug that disabled the functions (such as `NNS_G2dSetSpriteCurrentPolygonID()`) used to set polygon IDs used for drawing with software sprites.
- `NNS_G2dDrawSprite*()` no longer internally emulates wrapping when an off-screen position is specified.

---

## 2.9.4 Changes Related to Graphical Image Loading

---

- Added a palette load function `NNS_G2dLoadPaletteEx()` that loads only the portion of the palette that is being used. To use this function, you must use the converter's `-pcm` option to create compressed palette data.
- When `NNS_G2dLoadPalette()` was used to load 256-color extended palette data for the 2D graphics engine, the loading address value was not reflected. This bug was fixed.

---

## 2.9.5 Renderer

---

- Added the `NNS_G2dSetRendererOverwriteEnable()`, `NNS_G2dSetRendererOverwrite*()`, and other functions that overwrite the drawing OBJ parameters and are easier for a programmer to manipulate.
- You can now control the OBJ affine transform mode overwrite methods, such as `NNS_G2dSetRendererAffineOverwriteMode()`, used when the renderer draws OBJs with the help of the 2D graphics engine.
- Changed the argument type used by callbacks before and after OBJ is drawn.
- Fixed the bug because of which cells meeting both of the following conditions were not drawn properly:
  - The load address of the image proxy character data is a non-zero value.
  - The character data use a 256-color palette format and do not use a 1D-32K mapping mode.

---

## 2.9.6 Converter

---

- Added the `-pcm` option. This option outputs compressed palette data that store only the used palette numbers as data. Palettes can be partially loaded using `NNS_G2dLoadPaletteEx()` to load the compressed palette data.
- Output has been changed to standard error output. Also, the user can now control the output of the converter's operation messages. By default, the converter does not output operation messages. If option `-v` is used, the operation messages are sent to standard output.
- The value for the converter's execution result is now returned. If execution is successful, 0 is returned. If not successful, a non-zero value is returned.
- If object compression was turned OFF during the conversion of 256-color palettes and 1D mapping characters, a bug caused the exported data to become corrupted. This bug was fixed.
- Added the `-br` option, which outputs the cell data that contain rectangular area information.

---

## 2.10 Changes from Version 0.8.0

---

### 2.10.1 Overall

---

Added the feature that displays BG semi-automatically, using screen data information.

### 2.10.2 OAM Manager

---

Changed the way in which double-size affine transformation mode is applied forcibly to an OBJ saved using `NNS_G2dEntryOamManagerOamWithAffineIdx()`. Only the affine parameter numbers are rewritten. The affine transformation mode no longer changes.

The same processing must be performed outside of the manager.

**Note:** When using `NNS_G2dMakeCellToOams()` and the renderer, you do not need to be concerned with this. In the next release, we plan to release an API that controls the rewriting operations performed in the renderer's affine transformation mode.

### 2.10.3 Extended OAM Manager

---

We removed the feature for setting the OBJ affine transformation mode (normal or double-size). As a result, the `NNS_G2dSetOamManExDoubleAffineFlag()` and `NNS_G2dGetOamManExDoubleAffineFlag()` functions were deleted. The same processing must be performed outside of the manager.

**Note:** You do not need to be concerned with this when using `NNS_G2dMakeCellToOams()` and the renderer.

### 2.10.4 Renderer

---

Implemented the processing that supports the switching of the active state of the render surface. This feature allows specific surfaces to forcibly disable drawing. Added the `NNS_G2dSetRendererSurfaceActiveFlag()` and `NNS_G2dGetRendererSurfaceActiveFlag()` accessor functions.

### 2.10.5 Converter

---

Added support for the conversion of BG data.

---

## 2.11 Changes from Version 0.7.0

---

### 2.11.1 OAM Manager

---

Added a feature (`NNS_G2dSetOamManagerSpriteZoffsetStep()`) to shift the z value automatically and to draw sprites with the software sprite drawing feature.

### 2.11.2 Renderer

---

- Added the `NNS_G2dSetOamManagerSpriteZoffsetStep()` function, which shifts the z value automatically and draws sprites when the software sprite drawing feature is used.
- Examined the renderer setting status and implemented the `NNS_G2dOptimizeRenderer()` function, which optimizes the rendering process.
- Implemented the flip drawing function `NNS_G2dSetRendererFlipMode()`.
- Made it possible to register the callback functions that are used before and after drawing a cell or an OBJ.

### 2.11.3 Cell Animation

---

Added support for SRT (scale, rotate, and translate) animation.

**Note:** Many of the NCE data that were created with the previous versions of NITRO-CHARACTER contain animation frames that include an incorrect scale zero value. Such data cannot be drawn properly with the new runtime. Reconvert the data and replace the scale zero, or save the data again with the latest version of NITRO-CHARACTER. (This applies to the multicell animation described below.)

### 2.11.4 Multicell Animation

---

- Added the support for SRT (scale, rotate, and translate) animation.
- Fixed the bug that did not reset the cell animation playback status when multicells were switched. Now the multicell animations that are set to STOP mode can be played back properly.

### 2.11.5 Converter

---

Implemented the feature that replaces with “1” the scale value specified as “0” for the SRT animation data.

---

## 2.12 Changes from Version 0.6.0

---

### 2.12.1 Animation

---

- Fixed the problem that caused the animation frame of frame 0 to be set regardless of the frame number entered with `NNS_G2dSetAnimCtrlCurrentFrame()`.
- Fixed problems with:
  - `NNS_G2dSetMCAnimationCurrentFrame()`.
  - `NNS_G2dSetCellAnimationCurrentFrame()`.
  - `NNS_G2dSetCellAnimationCurrentFrameNoReset()`.

Internally, all these functions call `NNS_G2dSetAnimCtrlCurrentFrame()`.

### 2.12.2 Converters

---

- Fixed the bug that caused parameter interpretation to fail with the file names that do not include a `'\'` or `'/'` in their file path.
- Added the output option `-bg` for character (`ncg`) data that used BG. When using this option, no `ncc` (cell definition) file is necessary during conversion.
- Added option `-lbl`, which outputs the label name definition header file. The label name definition header file defines label text strings linked to animation sequences as aliases for label numbers.
- The converter log message was changed to output more data.

## 2.13 Changes from Version 0.5.0

### 2.13.1 General

- Changed the name of the converter program from `G2DConv.exe` to `g2dcvtr.exe`, and expanded the features. For details, see the converter manual.
- Changed names for the functions listed in the tables below. Function names used in the previous versions have been aliased and still work. For new projects, use the new names.

#### Animation

Old	New
<code>NNS_G2dGetCurrentElement</code>	<code>NNS_G2dGetAnimCtrlCurrentElement</code>
<code>NNS_G2dGetNextElement</code>	<code>NNS_G2dGetAnimCtrlNextElement</code>
<code>NNS_G2dGetNormalizedTime</code>	<code>NNS_G2dGetAnimCtrlNormalizedTime</code>
<code>NNS_G2dSetCallBackFuncutor</code>	<code>NNS_G2dSetAnimCtrlCallBackFuncutor</code>
<code>NNS_G2dSetCallBackFuncutorAtAnimFrame</code>	<code>NNS_G2dSetAnimCtrlCallBackFuncutorAtAnimFrame</code>
<code>NNS_G2dGetAnimSpeed</code>	<code>NNS_G2dGetAnimCtrlSpeed</code>
<code>NNS_G2dSetAnimSpeed</code>	<code>NNS_G2dSetAnimCtrlSpeed</code>
<code>NNS_G2dResetAnimationState</code>	<code>NNS_G2dResetAnimCtrlState</code>
<code>NNS_G2dInitCallBackFuncutor</code>	<code>NNS_G2dInitAnimCallBackFuncutor</code>
<code>NNS_G2dBindAnimController</code>	<code>NNS_G2dBindAnimCtrl</code>
<code>NNS_G2dGetAnimControllerType</code>	<code>NNS_G2dGetAnimCtrlType</code>
<code>NNS_G2dInitAnimController</code>	<code>NNS_G2dInitAnimCtrl</code>
<code>NNS_G2dIsAnimControllerActive</code>	<code>NNS_G2dIsAnimCtrlActive</code>
<code>NNS_G2dStartAnimController</code>	<code>NNS_G2dStartAnimCtrl</code>
<code>NNS_G2dStopAnimController</code>	<code>NNS_G2dStopAnimCtrl</code>
<code>NNS_G2dTickAnimController</code>	<code>NNS_G2dTickAnimCtrl</code>
<code>NNS_G2dInitAnimControllerCallBackFuncutor</code>	<code>NNS_G2dInitAnimCtrlCallBackFuncutor</code>

#### Cell Animation

Old	New
<code>NNS_G2dSetCellAnimSpeed</code>	<code>NNS_G2dSetCellAnimationSpeed</code>
<code>NNS_G2dGetCellAnimAnimCtrl</code>	<code>NNS_G2dGetCellAnimationAnimCtrl</code>
<code>NNS_G2dInitializeCellAnimation</code>	<code>NNS_G2dInitCellAnimation</code>
<code>NNS_G2dInitializeCellAnimationVramTransferred</code>	<code>NNS_G2dInitCellAnimationVramTransferred</code>



### Multicell Animation

Old	New
NNS_G2dInitializeMCAnimation	NNS_G2dInitMCAnimation
NNS_G2dInitializeMCInstance	NNS_G2dInitMCInstance
NNS_G2dSetMCAnimSpeed	NNS_G2dSetMCAnimationSpeed

### Picture Image Loading

Old	New
NNS_G2dInitializeImageProxy	NNS_G2dInitImageProxy
NNS_G2dInitializeImagePaletteProxy	NNS_G2dInitImagePaletteProxy

### Entity

Old	New
NNS_G2dInitializeEntity	NNS_G2dInitEntity
NNS_G2dSetCurrentAnimation	NNS_G2dSetEntityCurrentAnimation

### OAM Manager

Old	New
NNS_G2dInitializeOamManager	NNS_G2dInitOamManagerModule
NNS_G2dEntryNewOam	NNS_G2dEntryOamManagerOam
NNS_G2dEntryNewOamWithAffineIdx	NNS_G2dEntryOamManagerOamWithAffineIdx
NNS_G2dEntryNewOamAffine	NNS_G2dEntryOamManagerAffine
NNS_G2dEntryOamAffine	NNS_G2dSetOamManagerAffine
NNS_G2dApplyToHW	NNS_G2dApplyOamManagerToHW
NNS_G2dApplyToHWSoftEmu	NNS_G2dApplyOamManagerToHWSprite
NNS_G2dResetBuffer	NNS_G2dResetOamManagerBuffer
NNS_G2dApplyToHWAndReset	NNS_G2dApplyAndResetOamManagerBuffer
NNS_G2dGetOamAttrCapacity	NNS_G2dGetOamManagerOamCapacity
NNS_G2dGetOamAffineCapacity	NNS_G2dGetOamManagerAffineCapacity

**Expanded OAM Manager**

Old	New
NNS_G2dEntryNewOamEx	NNS_G2dEntryOamManExOam
NNS_G2dEntryNewOamWithAffineIdxEx	NNS_G2dEntryOamManExOamWithAffineIdx
NNS_G2dEntryNewAffineParamsEx	NNS_G2dEntryOamManExAffine
NNS_G2dApplyToBaseModuleEx	NNS_G2dApplyOamManExToBaseModule
NNS_G2dResetOamInstanceEx	NNS_G2dResetOamManExBuffer
NNS_G2dGetNewOamInstanceEx	NNS_G2dGetOamManExInstance
NNS_G2dSetOamEntryFunctionsEx	NNS_G2dSetOamManExEntryFunctions

**Renderer**

Old	New
NNS_G2dAddTargetSurface	NNS_G2dAddRendererTargetSurface
NNS_G2dSetCurrentImageProxy	NNS_G2dSetRendererImageProxy

**Software Sprites**

Old	New
NNS_G2dSetCurrentPolygonID	NNS_G2dSetSpriteCurrentPolygonID
NNS_G2dSetDefaultAttr	NNS_G2dSetSpriteDefaultAttr
NNS_G2dGetDefaultAttr	NNS_G2dGetSpriteDefaultAttr

**2.13.2 Loading Files**

Improved debug character string output of the color palette.

**2.13.3 OAM Manager**

- Added `NNS_G2dGetNewOamManagerInstance()`, a new function that generates the OAM Manager entity.
- Added `NNS_G2dGetNewOamManagerInstanceAsFastTransferMode()`. This function creates the OAM Manager entity and, using DMA, can transfer the internal buffer at a high speed.
- Added support for the cases in which the OAM Manager does not control the affine parameter.
- In the internal buffer reset process, fixed the problem where affine parameters outside of the control range were incorrectly reset by `NNS_G2dResetBuffer()`.
- Fixed the problem of incorrect drawing when the software sprite drawing function is used outside of double affine mode.

**2.13.4 Software Sprites**

- Changed the parameters of the 3D graphics engine camera settings for software sprites.
- Fixed the problem with the disparities in the image output between 2D and 3D drawing.

---

## 2.14 Changes from Version 0.4.0

---

### 2.14.1 General

---

- Added support for multicell.
- Changed the version of every file format to 1.0. Note that the old data cannot be used. Use the new converter to reconvert.
- Added cell VRAM transfer state manager and module. These are used in VRAM transfer animation.
- From OAM Manager, separated the drawing function that uses OAM software sprites. Added the OAM software sprite drawing module.

### 2.14.2 Cell Animation

---

Added support for VRAM transfer animation.

### 2.14.3 Multicell

---

- Changed the name of the `NNS_G2dGetNumNodesRequired()` function to `NNS_G2dGetMCNumNodesRequired()`.
- Changed the name of the `MultiCellDataToMCInstance()` function to `NNS_G2dSetMCDataToMCInstance()`.

### 2.14.4 OAM Manager

---

- Changed the drawing function that uses the OAM software sprites to be a separate module.
- Changed the behavior of `NNS_G2dDrawOneOam3DDirect()` accordingly. For details, see the API reference.

### 2.14.5 Converter

---

- Implemented version checking for NITRO-CHARACTER files. Old version files can no longer be used. That is, old data must be resaved using the new NITRO-CHARACTER.
- Added various output options. For details, see the converter documentation.
- Added support for the output of various 1D mapping modes.
- Added support for converting the data that use the character compression function.
- Enabled the output of the VRAM transfer animation data.
- Added support for the output of the animation play format information.

## 2.15 Changes from Version 0.3.0

---

### 2.15.1 General

---

- Combined the Extended OAM Manager and the Extended OAM Manager 2 into the Extended OAM Manager. The API uses the Extended OAM Manager 2 format.
- Changed the format of the `Cell` definition data NCER file. Data from previous versions cannot be used, and re-conversion is necessary.
- Renamed `NNSG2dCellData.cellType` to `NNSG2dCellData.cellAttr`.
- The function group that outputs runtime library debug data is defined as a dummy inline function in the FINALROM build.

### 2.15.2 Animation Controller

---

Converted the accessor group functions that were defined in the header to inline functions.

### 2.15.3 Extended OAM Manager

---

- Fixed the bug that caused improper rendering when OAM was registered without using affine parameters.
- Can specify the method as normal or double-sized affine for affine transformations that are used by the OAM Manager. (`NNS_G2dSetOamManExDoubleAffineFlag()`)
- Can no longer configure the default rendering registration functions. Must execute the `NNS_G2dSetOamEntryFunctionsEx()` function.

### 2.15.4 Renderer

---

- Added a Visible Culling feature to the Renderer. Data for the new converter must be reconverted.
- Corrected the problem that prevented proper rendering when the OBJ position was  $y \leq -64$  in the local coordinate system in `Cell`.
- Added the `NNS_G2dInitRenderSurface()` function to initialize the render surface.
- Added a pointer to the function that performed Visible Culling on the render surface member.

---

## 2.16 Changes from Version 0.2.0

---

### 2.16.1 File Loading

---

When `NNS_G2dPrintAnimContents()` was executed, a bug caused the debug output to fail for the animation data that uses SRT (Scale, Rotate, and Translate) animation.

### 2.16.2 OAM Manager

---

- Fixed the bug that caused improper operation when `NNS_G2D_OAMTYPE_SOFTWAREEMULATION` was specified as a manager type, and the Oam management region start index was specified as a value other than 0.
- Fixed the bug that caused improper operation when the affine parameter management region start index was specified as a value other than 0.

### 2.16.3 Software Sprite

---

- Fixed the bug that caused improper drawing after a sprite with an alpha value of 0 was drawn.
- Fixed the bug that caused improper drawing when the `NNS_G2D_SPRITEATTR_UV` attribute was not used.
- For the off-screen software sprites, corrected the problem of a difference in the behavior of the 2D graphics engine that arose with the position turnaround process.

### 2.16.4 General

---

To allow multiple OBJs in a single `Cell` to share the character region during 1D mapping, added converter support to the specifications.

---

## 2.17 Changes from Version 0.1.0

---

### 2.17.1 File Loading

---

- Added the `NNS_G2dGetUnpackedXXX()` function group to simplify the writing of user code.
- Added the `NNS_G2dIsBinFileValid()` function to verify file extensions and versions.

### 2.17.2 Animation Controller

---

- Added playback control functions, including `NNS_G2dStartAnimController()` and `NNS_G2dStopAnimController()`.
- To allow the programmer to change the playback mode, added `NNS_G2dSetAnimCtrlPlayModeOverridden()` and other functions.
- Renamed `NNS_G2dSetCallBackFunctorAtSpecifiedFrame()` to `NNS_G2dSetCallBackFunctorAtAnimFrame()`.

---

### 2.17.3 OAM Manager

---

Changed the arguments for `NNS_G2dApplyToHWSoftEmu()` and `NNS_G2dDrawOneOam3DDirect()`.

---

### 2.17.4 Extended OAM Manager 2

---

Changed the arguments for `NNS_G2dGetNewOamInstanceEx2()`.

---

### 2.17.5 Changes Related to Graphical Image Load

---

Added a group of convenience functions to simplify data downloads to an image proxy.

---

### 2.17.6 Entity

---

Provided the `BuildNENR.exe` tool to create entity definition binary files.

---

### 2.17.7 Renderer

---

- Changed the arguments for drawing related methods. Hid the matrix cache for the `NNS_G2dDrawXXX` internal data structure.
- Changed the arguments for `NNS_G2dPopMtx()`.

---

### 2.17.8 Sample

---

- Revised to allow the loading of sample data from the file system that supports NITRO-SDK 1.2.
- Added character data display samples for each format, including the 256-color palette and the extended palette.

---

### 2.17.9 General

---

- Allowed the compiler switch to turn on/off the debug output in the library. Debug output is enabled when `NNSI_G2D_DEBUG` is defined.
- Changed the file format and the converter. To use previously created files, reconvert them.

## 3 Known Problems

No problems have been reported at this time.

## 4 Future Plans

There are no future plans at this time.

© 2004-2007 Nintendo

No part of the contents of this document may be reproduced, copied, transferred, distributed, or given without the permission from Nintendo.