

## 目次

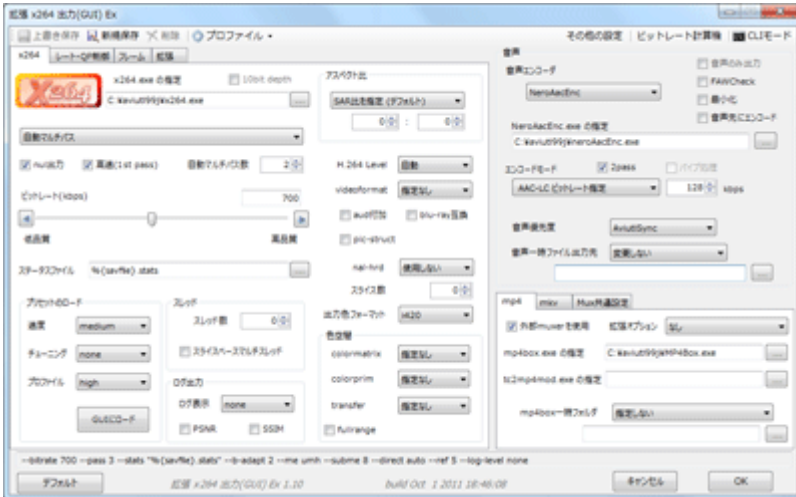
- [総説](#)
- [サーバサイドエンコードについて](#)
- [MPEG-4 AVC / H.264とは](#)
- [用意するもの](#)
  - [x264](#)
  - [拡張 x264 出力\(GUI\) Ex](#)
  - [Nero AAC Codec](#)
  - [MP4Box](#)
- [実行ファイルの指定](#)
- [x264guiExの簡単な使い方](#)
- [x264guiExの設定](#)
  - [「x264」タブ](#)
  - [「レート・QP制御」タブ](#)
  - [「フレーム」タブ](#)
  - [「拡張」タブ](#)
  - [音声](#)
- [リサイズについて](#)
  - [リサイズする理由](#)
  - [リサイズの具体例](#)
  - [リサイズの方法](#)
- [高画質な動画にするためのポイント](#)
  - [x264guiExの設定を見直す](#)
  - [適切なソースを選ぶ](#)
  - [映像ビットレートを高くする](#)
  - [エコノミーモードを回避する](#)
- [こんなときは](#)
  - [動画を再生するとPCが重くなる](#)
  - [エンコード時間を短くしたい](#)
  - [映像が単一の色で乱れる](#)
- [その他](#)
- [関連ページ](#)

---

## 総説

- このページでは、動画編集ソフトである [AviUtil](#) を使って [MPEG-4 AVC / H.264](#) というファイル形式の動画を作成する方法について解説しています。AviUtilのことを知らない、同ソフトを使ったことがないという方は解説を読む必要はありません。また、AviUtilで読み込んだ動画が

すでに完成している（編集が終了している）ことを前提に解説しています。AviUtlでの動画編集の方法については、[AviUtlの使い方](#)をご覧ください。



AviUtlでH.264形式の動画を作成するさいに使うプラグインの設定画面

- ニコニコ動画に動画を投稿するうえで気をつけたいポイントはふたつあります。すなわち、（1）AviUtlでエンコードした動画をそのまま投稿すると、ニコニコ動画側で再エンコードされるという点と、（2）動画投稿時にエラーが出て投稿できないことがある、という点のふたつです。この2点はとても重要なので、しっかりと理解しておきましょう。
- （1）は、動画投稿時にニコニコ動画側で強制的に動画が再エンコードされることによって、画質・音質がきわめて劣化することを意味しています。また（2）は、動画を投稿しようとしたさいに、ファイルサイズが大きいなどのエラーが表示されることがあるということです。そこで、このページでは AviUtlを使用して、ニコニコ動画に投稿できる高画質・高音質な動画を出力するための方法を見ていきます。
- なお、以下の解説が難しい場合は、ニコエンコまたはつんでれんこを使うとよいでしょう。

画面の上へ

## サーバサイドエンコードについて

- いま述べた（1）について述べておきますが、一定の要件を満たしていない動画はニコニコ動画に動画を投稿するさい、同サイトによって 強制的に動画が再エンコードされます。このことを サーバサイドエンコード といいます。「ニコニコ動画に動画を投稿したら画質が悪くなった」という人がいますが、理由は動画がサーバサイドエンコードされたからです。同エンコードによって、画質が予想以上に落ちることがあるのです。

動画投稿前



動画投稿後



- しかし逆に、[一定の要件さえ満たしていればサーバーサイドエンコードは回避することが可能](#)です。つまり、AviUtilで出力した動画をそのままの品質で投稿できるわけです。サーバーサイドエンコードを回避して動画を投稿するための要件をまとめたのが下表です。難しい用語があるかもしれませんが、簡単に見ておきましょう。

コンテナ	MP4
映像コーデック	MPEG-4 AVC / H.264
画面サイズ (解像度)	プレミアム会員 制限なし
	一般会員 1280x720以内
フレームレート	0.33fps ~ 120fps
音声コーデック	MP3, PCM AAC
サンプリングレート	22kHz, 44.1kHz 48kHz以下
ファイルサイズ	プレミアム会員 100MB以下
	一般会員 40MB以下
ビットレート (映像+音声)	プレミアム会員 制限なし
	一般会員 600kbps以下

[画面の上へ](#)

## MPEG-4 AVC / H.264とは

- ここで、上表の「コーデック」のところで登場したMPEG-4 AVC / H.264について簡単に見ておきましょう。[MPEG-4 AVC / H.264](#)は、動画の圧縮規格のひとつです。従来の圧縮規格であるMPEG-2よりも高画質でファイルサイズの小さい動画を作成できます。MPEG-4 AVC / H.264は、たんにMPEG-4 AVCまたは[H.264](#)ともいいます。



H.264は、MPEG-2やDivXなど他の動画圧縮技術と比較して高画質・高圧縮である点に特徴があります。数ある動画圧縮規格のうち、現在は主流のひとつです。

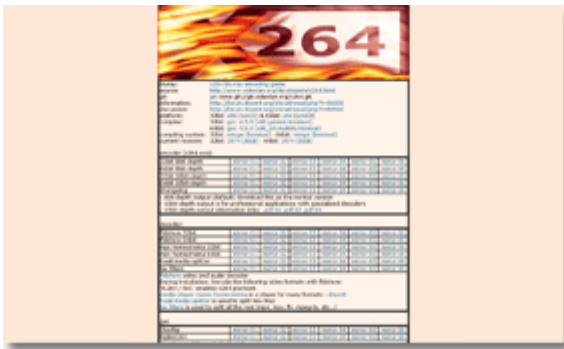
- 用語が難しくても心配いりません。重要なのは、[H.264とよばれる動画をニコニコ動画用にAviUtilで出力する](#)という点です\*1。ただ、AviUtilでH.264動画を出力するためには、[AviUtil以外にもソフトウェアを用意する必要があります](#)。また、[設定もニコニコ動画用に自分で変更しなくてははいけません](#)。したがって、あまり初心者向けの方法とはいえないのですが、このページではできるだけわかりやすく解説していきます。わかりづらいようであれば、[ニコエンコで変換](#)を読むようにしてください。

## 用意するもの

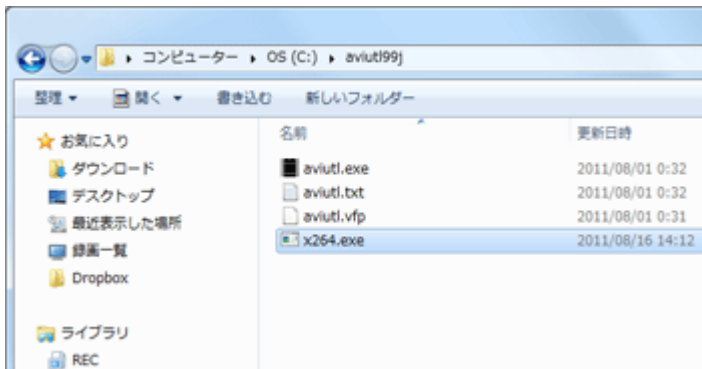
- AviUtlでH.264動画を出力するために必要なものは以下のとおりです。すべて無料です。

### x264

- [x264](#) は、H.264動画を作成するためのソフトウェアです。同ソフトウェアは[X264.nl](#)でダウンロードできます。OS が32bit版である場合は「32bit 8bit-depth」を、64bit版である場合は「64bit 8bit-depth」をそれぞれクリックして「x264.exe」をダウンロードしましょう。ダウンロード先が「mirror 01」から「mirror 06」までありますが、どれでもかまいません。



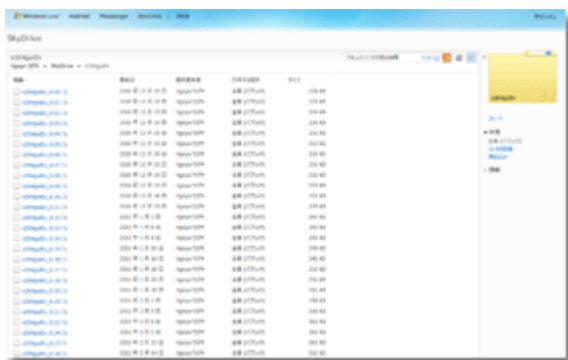
- そして、ダウンロードした「[x264.exe](#)」を適当な場所に置いておきます。場所はどこでもよいのですが、「aviutl.exe」と同じ場所に置いておくともわかりやすいかもしれません。「x264.exe」の場所は、あとで指定することになるので覚えておきましょう。



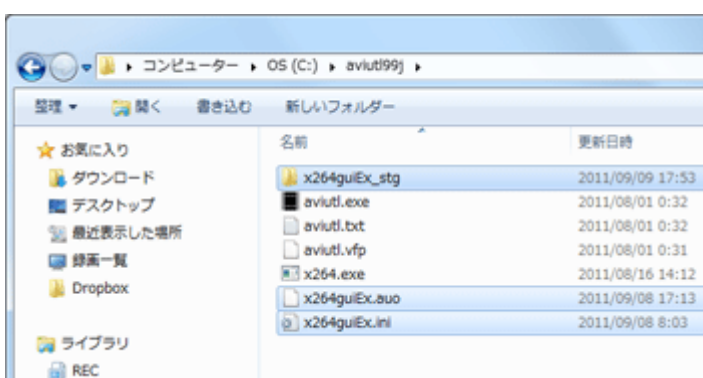
「x264.exe」を「aviutl.exe」と同じ場所に入れたところ

### 拡張 x264 出力(GUI) Ex

- [拡張 x264 出力\(GUI\) Ex](#) (以下、x264guiEx) は、x264を簡単な操作で扱えるようにしたAviUtl用プラグインです。x264guiExは[こちら](#)でダウンロードできます。最新版のx264guiExは、画面を下にスクロールしたところにあります。



- そして、ダウンロードした「x264guiEx\_ . . . .zip」を解凍します。すると、「x264guiEx\_ . . . .」フォルダが生成されます。このフォルダのなかに「auo」フォルダがあり、「auo」フォルダ内には「x264guiEx\_stg」「x264guiEx.auo」「x264guiEx.ini」の3つがあります。これらを「aviutl.exe」がある場所に置きます。



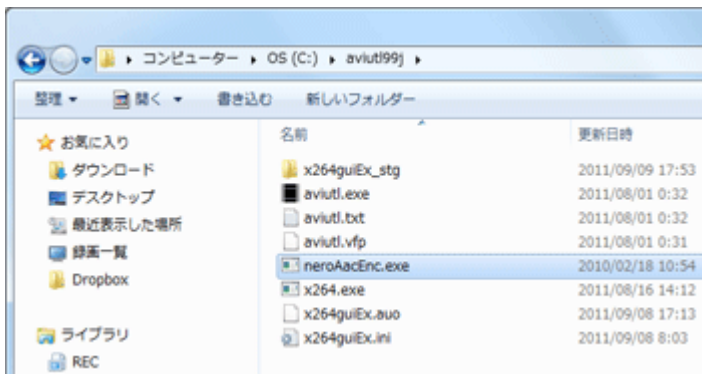
「x264guiEx\_stg」「x264guiEx.auo」「x264guiEx.ini」を「aviutl.exe」と同じ場所に入れたところ

## Nero AAC Codec

- [Nero AAC Codec](#) は、音声をAACという圧縮形式にエンコードするための コーデック です。Nero AAC Codecは、[Nero](#)からダウンロードできます。「同意する」をクリックして、メールアドレスを入力すればダウンロード可能な状態になります。



- そして、ダウンロードした「NeroAACCodec- . . . .zip」を解凍します。解凍してできたフォルダ内には「win32」フォルダがあります。このフォルダのなかにある「neroAacEnc.exe」を適当な場所に置いておきましょう。「aviutl.exe」と同じ場所に置いておくとわかりやすいかもしれませんが、「neroAacEnc.exe」の場所は、あとで指定するので覚えておきます。



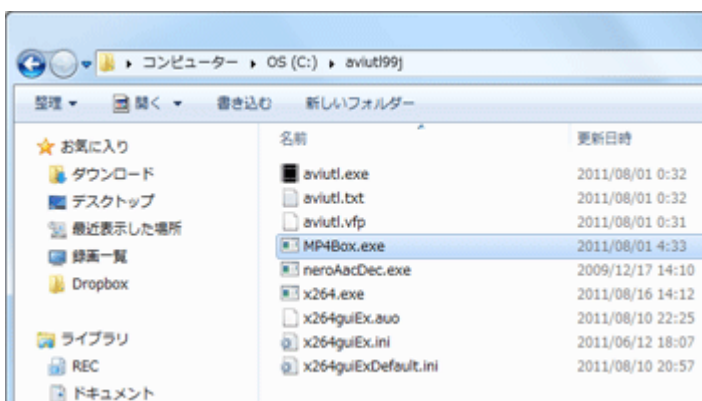
「neroAacEnc.exe」を「aviutl.exe」と同じ場所に入れたところ

## MP4Box

- [MP4Box](#) は、エンコードされた映像と音声を合成して、ひとつの動画ファイル（MP4ファイル / MP4 コンテナ）にするために必要です。MP4Boxは、[こちら](#)でダウンロードできます。最新版をダウンロードすればよいでしょう。



- ダウンロードした「MP4Box.zip」を解凍します。そして、解凍してできた「MP4Box」フォルダ内にある「[MP4Box.exe](#)」を適当な場所に置いてください。あとで「MP4Box.exe」の場所を指定するので覚えておきます。



「MP4Box.exe」を「aviutl.exe」と同じ場所に入れたところ

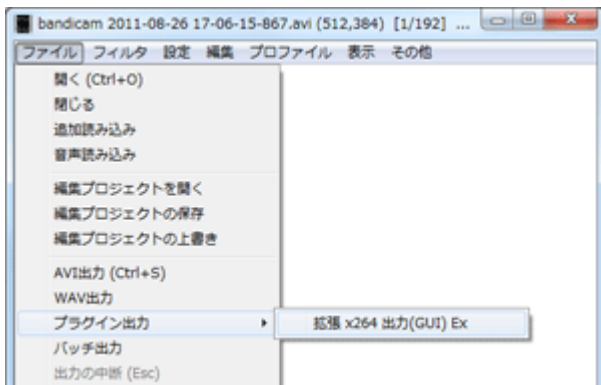
画面の上へ

## 実行ファイルの指定

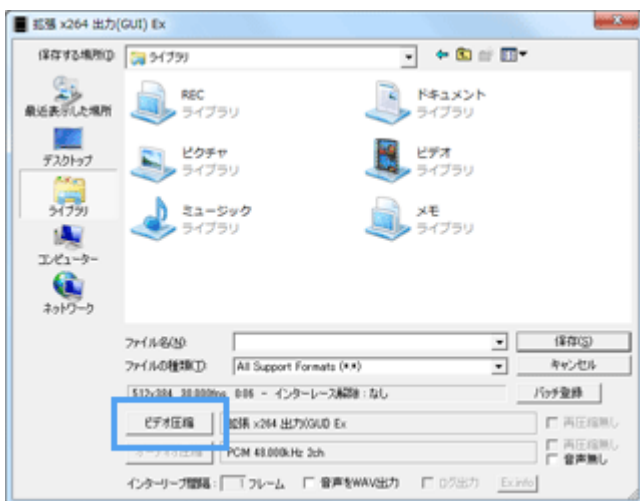
- いま用意した「x264.exe」「neroAacEnc.exe」「MP4Box.exe」を入れた場所を指定します。

いちど設定すれば次回以降は設定し直す必要はありません。

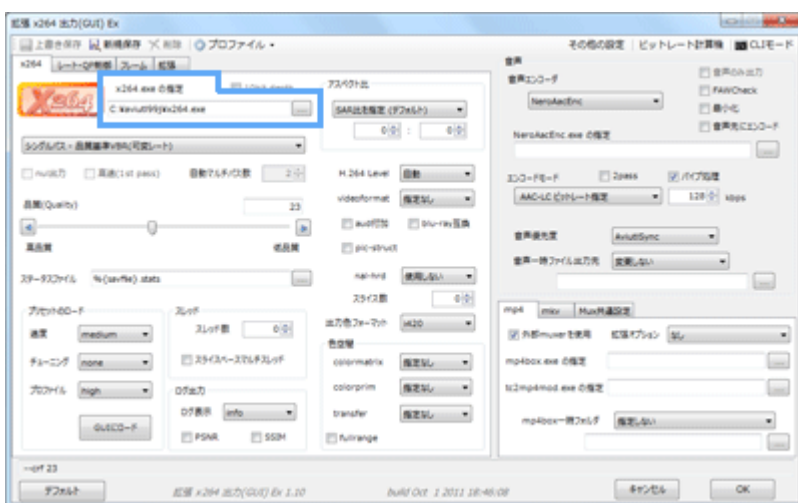
- 1 AviUtlで動画を開きます。
- 2 「ファイル」 「プラグイン出力」で「拡張 x264出力(GUI) Ex」を選択します。



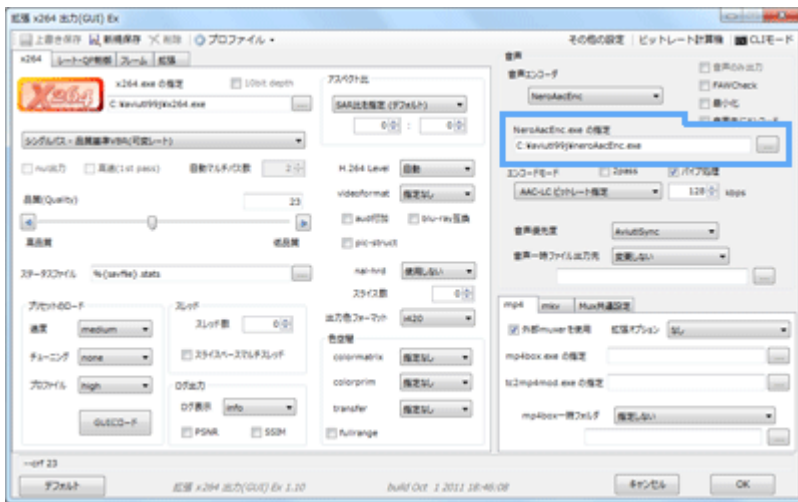
- 3 「ビデオ圧縮」をクリックします。



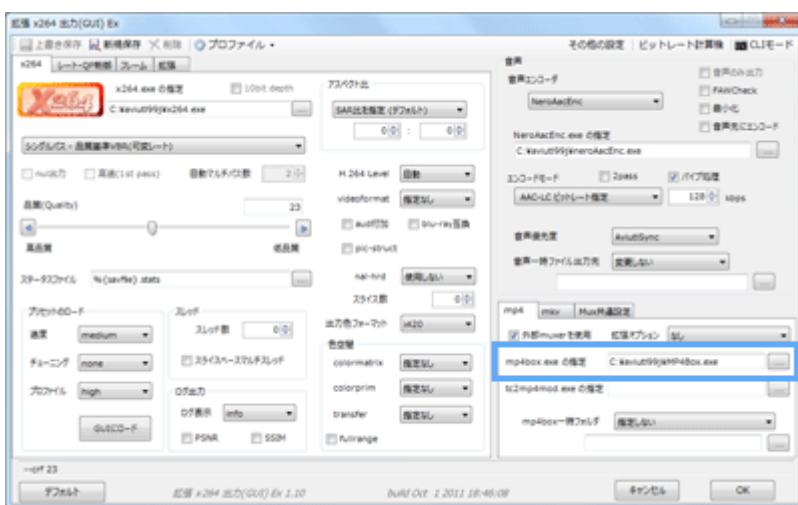
- 4 「x264」タブで「x264.exe」の場所を指定します。



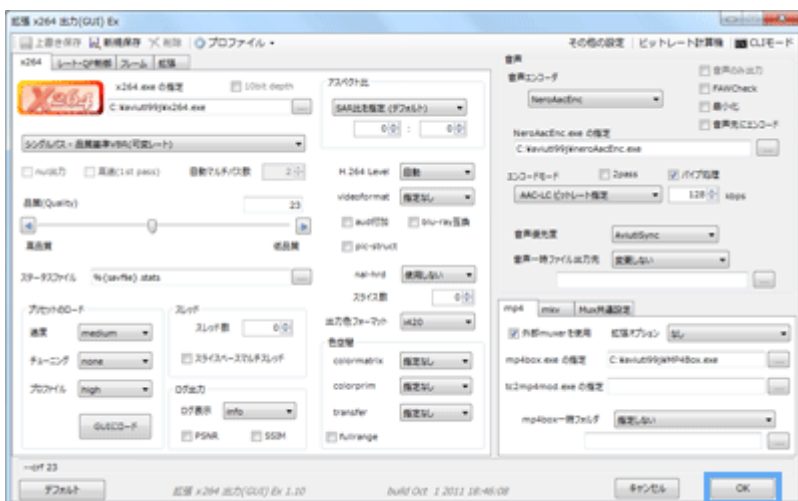
- 5 「音声」で「neroAACEnc.exe」の場所を指定します。



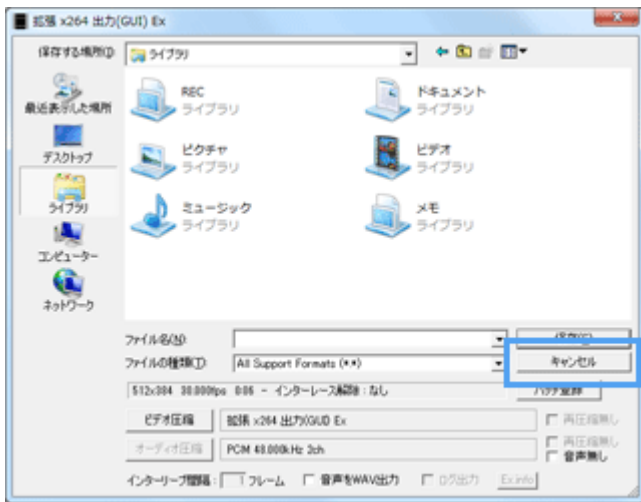
6 「.mp4」タブで「MP4Box.exe」の場所を指定します。



7 「OK」をクリックします。



8 「キャンセル」をクリックします。

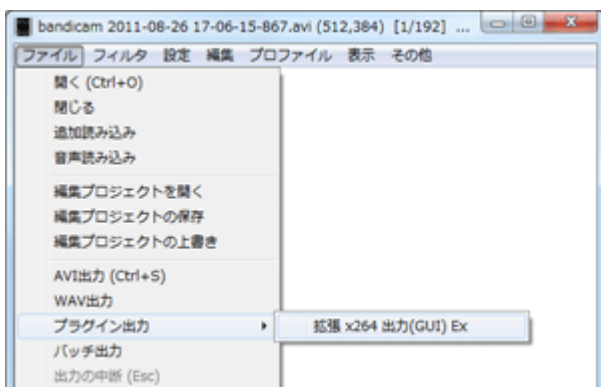


画面の上へ

## x264guiExの簡単な使い方

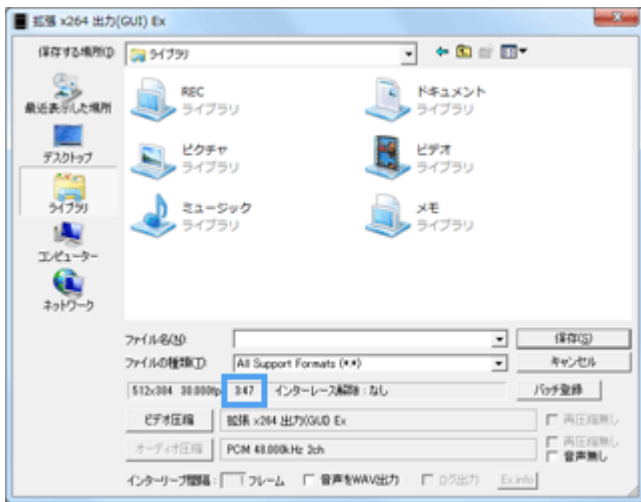
- ここまでできたら、再生時間が数分程度の短い動画を使ってエンコードしてみましょう。そして、実際に動画を投稿できるかテストします。

- 1 AviUtlで動画を開きます。動画編集などは事前に済ませておいてください。
- 2 ニコニコ動画のプレイヤーの画面サイズ（解像度）に合わせてリサイズ（縮小）します。動画の縦横比が4:3の場合は 512x384 に、動画の縦横比が16:9の場合は 640x360 にリサイズします。詳細は後述します。
- 3 「ファイル」 「プラグイン出力」で「[拡張 x264出力\(GUI\) Ex](#)」を選択します。

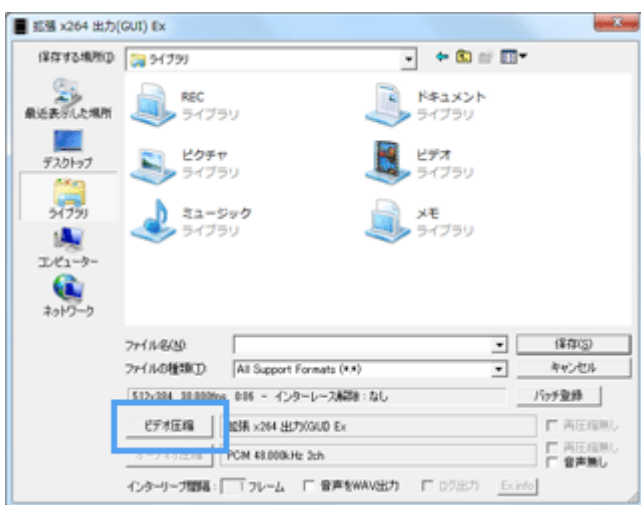


「ファイル」 「AVI出力」ではなく、「ファイル」 「プラグイン出力」 「[拡張 x264出力\(GUI\) Ex](#)」を選択します。

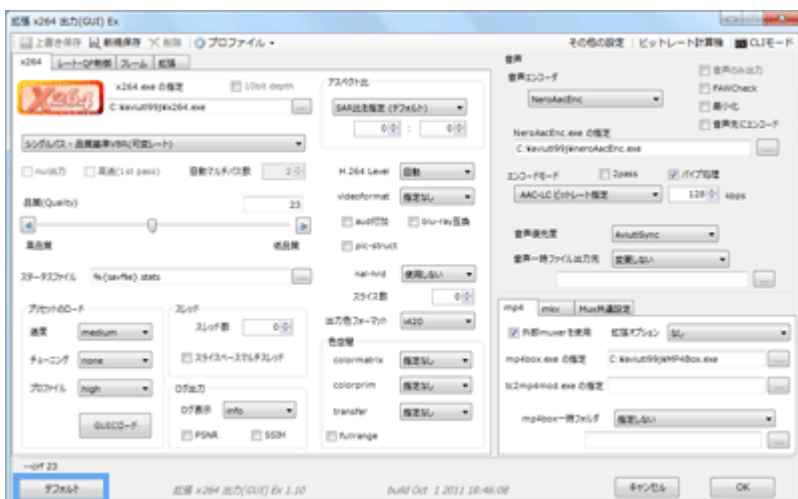
- 4 動画の再生時間を覚えておきます。あとで必要になります。



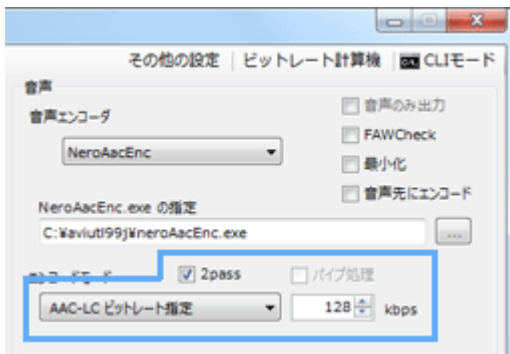
5 「ビデオ圧縮」をクリックします。



6 「デフォルト」をクリックします。今後、設定を変更したあとでも、同ボタンをクリックすることで初期設定に戻すことができます。



7 「AAC-LC ビットレート指定」を選択して「2pass」をONにします。音声ビットレートは任意の数値でかまいません（例：128kbps）。

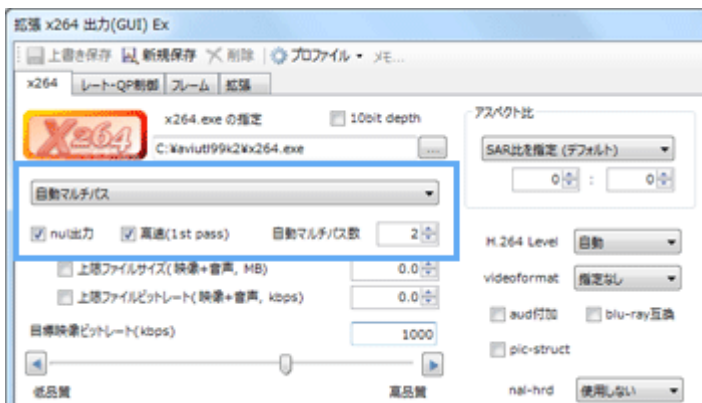


8 以下のように設定を変更します。この設定は自分で任意に変更してかまいません。あくまでも設定例ですので、参考程度に留めておいてください。

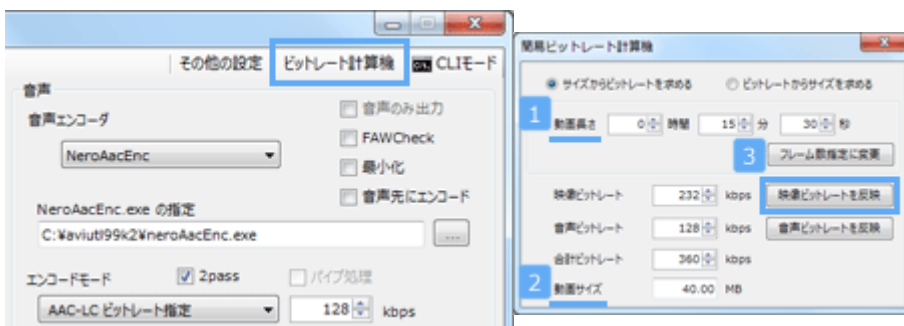


画像クリックで拡大。「x264」タブでは、「自動マルチパス」を選択し、「null出力」をON、「ログ表示」を「none」にしています。「フレーム」タブで「動き予測アルゴリズム」を「Uneven Multi-Hexagon」、「サブピクセル動き予測」を「8 (RD refinement for I/P frames)」、「動き予測方式」を「auto」、「参照距離」を5にしています。また、同タブで「適応的Bフレーム挿入」を「完全」にし、「キーフレーム間隔の上限」を300にしています。

9 「x264」タブで「自動マルチパス」を選択していることを確認してください。

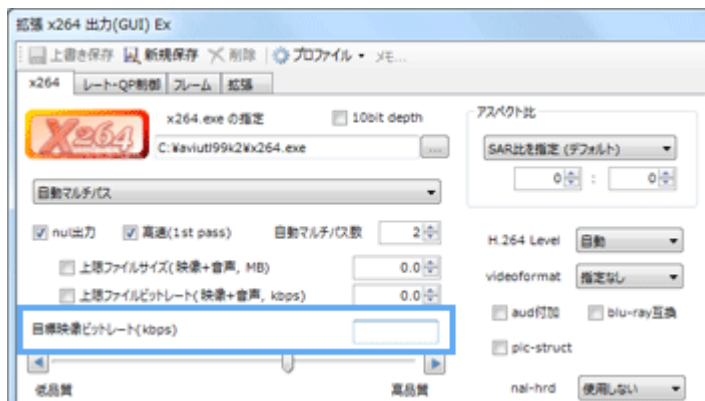


10 「ビットレート計算機」をクリックし、「動画長さ」に動画の再生時間を入力します。つぎに「動画サイズ」を、一般会員なら40MB、プレミアム会員なら100MBにします。そして、「映像ビットレートを反映」をクリックし、ビットレート計算機を閉じます。



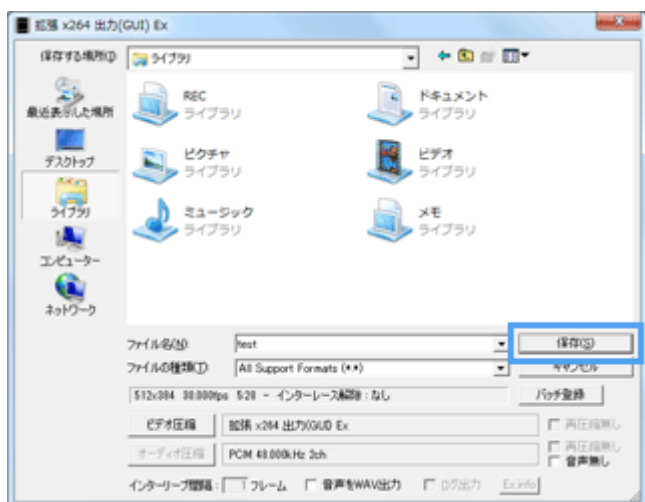
ビットレート計算機を使用して、映像に割り当て可能なビットレートを計算します。動画の再生時間や会員の種類により、映像に割り当てできるビットレートが異なるため、このような計算が必要になります。

11 一般会員の場合にかぎり、映像と音声の合計ビットレートが600kbps（あくまで目安）を超えた動画を投稿しようとすると、ニコニコ動画でエラーが表示されるため、「目標映像ビットレート」ではこの点に注意する必要があります \*2。



12 「OK」をクリックします。

13 適当なファイル名を入力し、動画の保存場所を決めて「保存」をクリックします。



14 エンコードが開始するので、しばらく待ちます。設定や動画の再生時間、PCのスペックによって待ち時間は異なります。

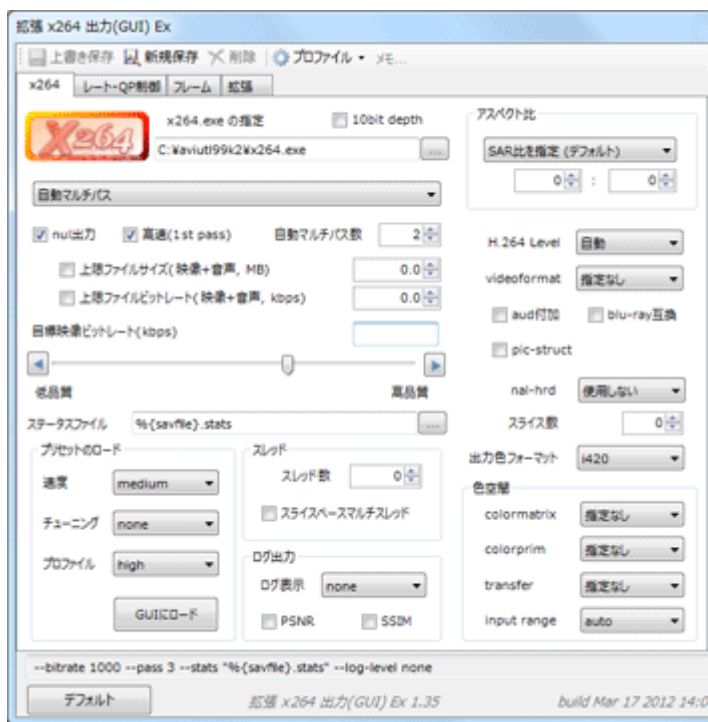
15 動画を投稿します。動画ファイル以外にステータスファイルができていますが（.statsおよびmbtree）、削除してかまいません \*3。

画面の上へ

## x264guiExの設定

- x264guiExでの設定について簡単に見ておきます。x264guiExの設定画面は、[左側が映像についての設定](#)、[右側が音声についての設定](#) というイメージで理解しておくとうわかりやすいでしょう。設定の詳細な意味については、[ニコニコ動画まとめwiki](#)を参考にしてください。なお、設定画面の「デフォルト」ボタンをクリックすることで設定を初期状態にできます。

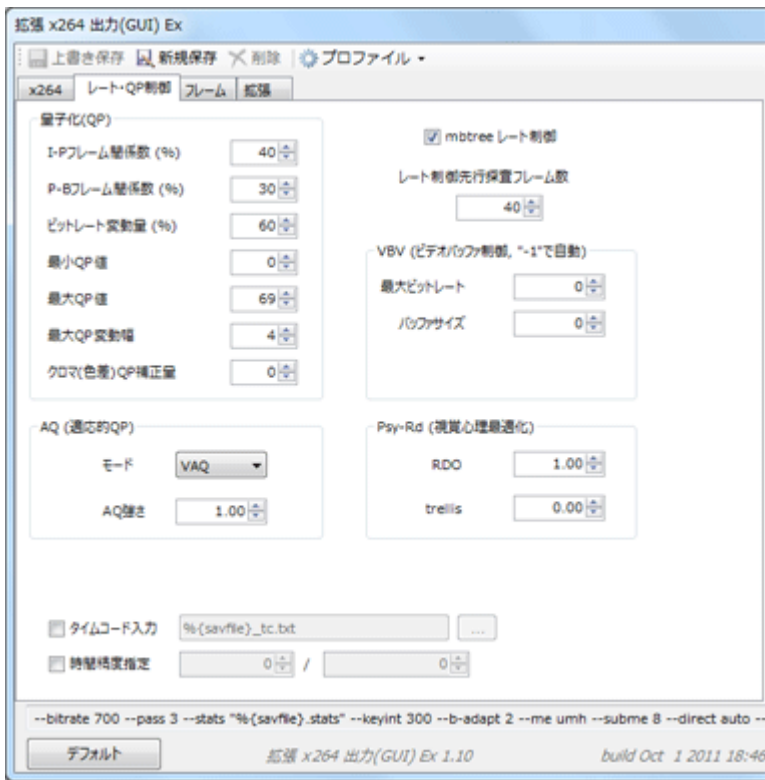
## 「x264」タブ



「x264」タブの設定例。「目標映像ビットレート」については、エンコードする動画ごとに毎回計算する必要があります。そのため、画像ではビットレートの入力欄を空白にしています。

- ニコニコ動画用の動画を出力する場合は、「自動マルチパス」または「シングルパス - ビットレート指定」を選択します。エンコード時間を短くしたいのであれば後者でもよいですが、どちらかというとな前者のほうがよいでしょう。自動マルチパスのほうが指定したとおりのビットレートでエンコードでき、かつ画質的にも有利だからです。
- たんにHDDに動画を保存しておく目的の場合は「シングルパス - 品質基準VBR（可変レート）」を選択すればよいのですが、これだとビットレートを指定できません。しかし、ニコニコ動画にはビットレートおよびファイルサイズにつき制限があるため、ビットレートは必ず指定する必要があります<sup>\*4</sup>。したがって、ニコニコ動画用の動画を出力する場合は「シングルパス - 品質基準VBR（可変レート）」を選択することはありません。
- ビットレートの算出方法については上で述べました。正しくビットレートを指定しても、設定によってはエンコード後の動画のビットレートおよびファイルサイズは目標値から多少前後するため、ビットレートの設定は余裕をもたせておくほうが無難です<sup>\*5</sup>。また、一般会員は映像と音声の合計ビットレートが600kbpsを超えないように注意しましょう。プレミアム会員であればこのような制限はありません。
- 投稿した動画をニコニコ動画で視聴したさいに、オリジナルの動画と色が違うと感じたときは色空間の問題が発生しているかもしれません。その場合は、「colormatrix」を「smpte170m」にします<sup>\*6</sup>。動画を投稿するまえに色を確認したい場合は、Flavieで再生してみましょ。色確認が正確にできます。

## 「レート・QP制御」タブ



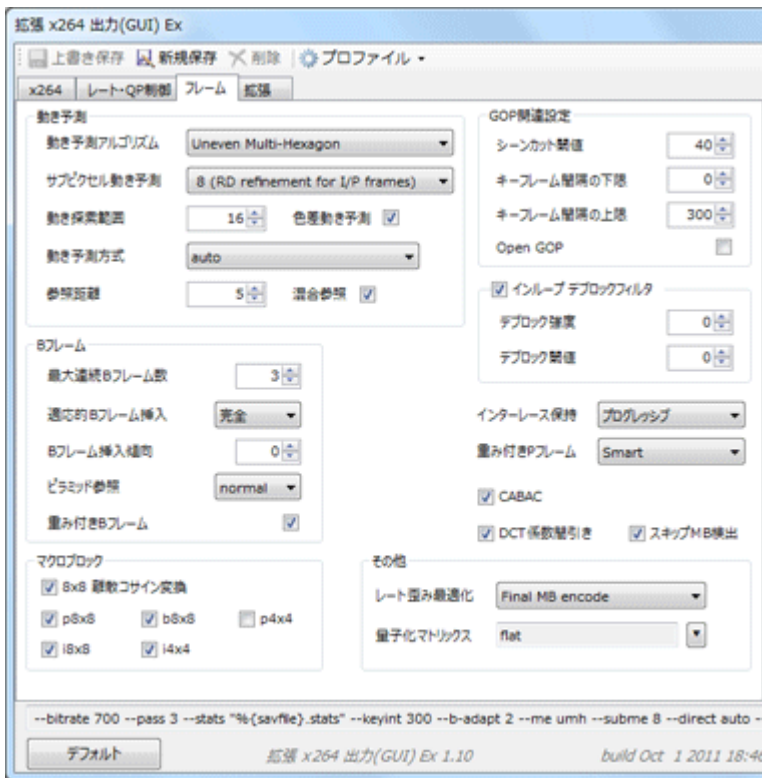
「レート・QP制御」タブの設定例

- 「レート・QP制御」タブで重要な項目は以下のとおりです。

#### 説明

AQ (適応的QP)	「VAQ」または「AutoVAQ」にすると画質が向上する。後者を選択して1.0~1.3にするとメリハリのある画質になることも。
Psy-Rd (視覚心理最適化)	基本的にデフォルトのままでよいが、場合によっては「RDO」を0~0.3にしたほうが画質がよいこともある。
mbtree レート制御	基本的にONでよい。

## 「フレーム」タブ



「フレーム」タブの設定例

- 「フレーム」タブで重要な項目は以下のとおりです。

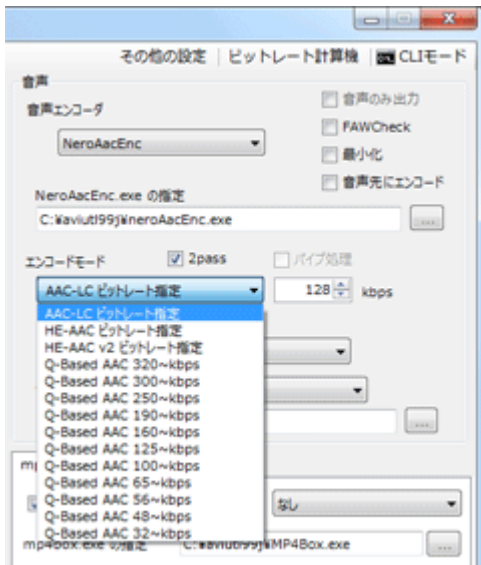
#### 説明

動き予測アルゴリズム	「Hexagonal Search」または「Uneven Multi-Hexagon」を選択。後者のほうが高画質だが、エンコード速度は遅くなる。
サブピクセル動き予測	数字が大きいほど画質が向上する。ただし、エンコード速度は遅くなる。7~9が使いやすい。
動き検索範囲	16または32にする。数値が大きいほうが圧縮率が向上する。ただし、エンコード速度は遅くなる。
動き予測方式	基本的に「auto」でよい。
参照距離	数値を大きくすると画質が向上し、圧縮率も向上する。再生負荷とエンコード時間を考えると3~6推奨。
最大連続Bフレーム数	数値が大きいほど高圧縮になる。通常は3~6にする。再生負荷が気になるときは3または4にする。
適応的Bフレーム挿入	「完全」にすると圧縮率が向上する。エンコード時間が気になるときは「簡易」にする。
重み付きBフレーム	ONでフェード時の画質が向上する。OFFで画質および再生負荷が下がる。基本的にON推奨。
マクロブロック	チェックが多いほど高画質。再生負荷が気になるときは「p4x4」のチェックを外す。
キーフレーム間隔の上限	10秒間隔のシーク移動でよいなら300、1秒間隔のシーク移動にしたいなら30（30fpsの場合）。300推奨。
インループ デブロックフィルタ	ONで画質が向上する。OFFで画質は下がるが、再生負荷は下がる。基本的にON推奨。-1で画像がシャープに。
CABAC	ONで画質が向上する。OFFで画質は下がるが、再生負荷は下がる。基本的にON推奨。
DCT係数間引き	基本的にONでよい。
スキップMB検出	基本的にONでよい。
レート歪み最適化	「Final MB encode」または「All」を選択することで画質向上。

## 「拡張」タブ

- 設定を変更する必要はありません。

## 音声



- 通常は、「AAC-LC ビットレート指定」を選択したうえで、音声ビットレートを112kbps、128kbps、144kbps、192kbpsのいずれかにすればよいでしょう。音声ビットレートを低くしたい場合は、「HE-AAC ビットレート指定」や「HE-AAC v2 ビットレート指定」から任意のものを選びます。具体的には、動画の再生時間が長い場合や、音質を犠牲にして映像に多くのビットレートを割り当てたい場合に、HE-AACやHE-AAC v2を選ぶと効果的です。

## 画面の上へ

## リサイズについて

### リサイズする理由

- **リサイズ**する意義として、(1) 画面の縦横比 (アスペクト比) を修正できる、(2) 動画再生時の CPU負荷 を軽減できる、(3) ビットレート が低いことによる画質破綻を防ぎやすい、(4) 一般会員は画面サイズが1280x720以内でないとサーバサイドエンコードされる、などがあげられます。(1) および (2) については、[AviUtilの使い方 / リサイズについて](#) で述べました。
- (3) は、**同一ビットレートの場合、リサイズした動画のほうがリサイズしていない動画よりもエンコード後の画質を維持しやすい** という意味合いです。たとえば、画面サイズが1280x720なのに映像ビットレートを600kbpsにしてエンコードすると、画質が破綻した動画になります。しかし、640x360にリサイズしておけば、同じ600kbpsでエンコードしてもそこそこの画質にはなります \*7。そこで、1280x720から640x360へリサイズするわけです。
- (4) はそのままの意味です。**一般会員の場合は、1280x720を超えた画面サイズの動画を投稿するとサーバサイドエンコードされてしまう** ため、動画投稿時に画質・音質が劣化してしまいます。そこで、1280x720以内にリサイズする必要があります。プレミアム会員の場合は、この

ような制限はありません。

## リサイズの実例

- では、実例を見ていきましょう。結論から述べると、**動画のアスペクト比が4:3の場合は512x384に、16:9の場合は640x360にリサイズ**するのが無難です。ニコニコ動画のプレイヤーの画面サイズは、512x384（4:3）または640x384（16:9）であるため、これに合わせるかたちになります。なお、2012年5月1日から開始したニコニコ動画の新バージョンである「Zero」では視聴画面サイズが大きくなりましたが、基本的に考慮する必要はありません。



4:3の動画を4:3モードで再生したとき（左）と、16:9の動画を16:9モードで再生したとき（右）の見え方（原宿バージョン）

- 動画のアスペクト比がわからない場合は、真空波動研などのソフトウェアを使えば容易に調べることができます。参考までに、動画のアスペクト比について下表にまとめました。ゲームを録画した場合を想定しています。たとえば、GV-USB2やSD-USB2CUP5などの従来型の ビデオキャプチャー で録画してできた動画の場合は、通常は512x384にリサイズすればよいでしょう\*8。

	画面サイズ	アスペクト比	指定するとよい画面サイズ
通常のキャプチャーボード	720x480	4:3 (3:2)	512x384 (4:3)
DVD / BDレコーダー	720x480	4:3 (3:2)	512x384 (4:3)
HDキャプチャーボード	1280x720など	16:9	640x360 (16:9)
PCゲーム (1)	800x600など	4:3	512x384 (4:3)
PCゲーム (2)	1280x720など	16:9	640x360 (16:9)

- 他方、HDビデオキャプチャー を使ってHD画質で録画した場合は、640x360にリサイズします。もっとも、640x360にリサイズせず、1280x720の画面サイズで動画を投稿してもかまいません。この場合、視聴者が画面を拡大したさいに美しい画質が楽しめるというメリットがあります。上述した(1)～(3)を考慮したうえでリサイズすべきかどうかを判断してください。

## リサイズの方法

- Lanczos 3-lobed 拡大縮小を使用している場合は、以下のようにしてリサイズします。

1. 「設定」 「サイズの変更」で「なし」になっていることを確認する。

「設定」 「Lanczos 3-lobed 拡大縮小の設定」の順にクリックする。

- 2.
3. 右上のチェックボックスにチェックを入れる。
4. 空欄に「512 x 384」または「640 x 360」と入力する。
5. 「指定」をクリックする。



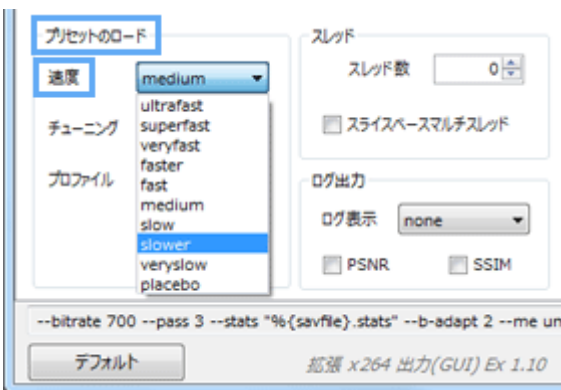
- もしリサイズ用のプラグインを使用していない場合は、「設定」 「サイズの変更」 「指定サイズ」で画面サイズを入力してください。また、[拡張編集プラグイン](#)を使用している場合のリサイズ方法については、[こちら](#)をご覧ください。
- アスペクト比には十分注意するようにしましょう。[アスペクト比がおかしいと、映像が縦または横方向に伸びた動画](#)になってしまいます。たとえば、720x480の動画を512x384にしないで512x288にリサイズすると、映像が横に伸びます。アスペクト比をまちがえたまま動画を投稿した場合、アスペクト比を修正するには動画をエンコードしなおし、再び投稿する必要があります。

画面の上へ

## 高画質な動画にするためのポイント

### x264guiExの設定を見直す

- x264guiExの設定を見直します。高画質な動画にするうえでこれがもっとも重要です。上表を参考に設定を変更してエンコードしてみてください。また、「x264」タブの「プリセットのロード」で、「速度」を「slow」または「slower」のいずれかにして「GUIにロード」ボタンをクリックすると、プリセットを使用できます。



下にあるプリセットほど高画質な設定です。その反面、エンコード時間は遅くなります。

- これらのプリセットを使用することで高画質な動画を簡単に作成できます。ただ、動画によって最適な設定は変わりますし、画質についての好みの問題もあるでしょう。また、高画質な動画にするための設定は、エンコード時間が長くなりがちです。したがって、けっきょくはどこまで動画ごとに最適な設定を見つけられるか、どこで妥協するのかという問題になります。

## 適切なソースを選ぶ

- AviUtlで読み込む動画（ソース）は高画質なものにしましょう。エンコードすれば画質は落ちます。エンコードすることで、もとの動画の情報が失われるからです。そのため、**低画質な動画をエンコードしても高画質にはできません**。より画質が落ちるだけです。したがって、高画質な動画をエンコードするようにしましょう。
- TVゲームを録画する場合、HDキャプチャーボードを使ってHD画質で録画すれば高画質な映像ソースとなります。PS3やXbox 360のゲームの録画にはHDキャプチャーボードが最適です。通常のキャプチャーボードで録画する場合であっても、ゲーム機とS端子接続して録画することで、わずかながら画質を向上させることが可能です。



DC-HC1（左）とHD PVR（右）

- エンコードに不向きな映像もあるので理解しておきましょう。たとえば、視点が頻繁に激しく動く動画や、登場キャラクターが画面内を縦横無尽に大きく動きまわる動画、細かく派手なエフェクトが連続して発生する動画などは、エンコードには向いていません。画質が劣化しやすくなります。

## 映像ビットレートを高くする

- 一般的に **映像ビットレートが高いほど高画質になります**。したがって、映像に多くのビットレートを割り当てるようにします。そのためには、（1）**動画の再生時間を短くする**、（2）**プレミアム会員になる**、というふたつの方法があります。
- （1）は、たとえば30分の動画と10分の動画を比較した場合、後者のほうが高ビットレートに設定できるので高画質にできるということです。ニコニコ動画の仕様によりH.264形式の動画は、一般会員は40MB、プレミアム会員は100MBまでとなっています。そして、ファイルサイズはビットレート×動画の再生時間で求められるので、**動画の再生時間が短いほうがビットレートを高くできる** わけです。動画の再生時間を短くするには、エンコードする範囲をAviUtlで短く設定するだけです \*9。
- （2）は、**プレミアム会員であれば一般会員とは異なりビットレート制限がない**ので、動画のファイルサイズが100MB以内であるかぎり、高ビットレートに設定できるということです。映

像ビットレートが高いほうが高画質になるわけですから、画質的にはプレミアム会員のほうが有利です。

## エコノミーモードを回避する

- エコノミーモードを回避する方法です。ただ、エコノミーモードを回避したほうがよいかどうかは、各人で判断してください。エコノミーモードで動画を再生したときの画質は、URLの末尾に「?eco=1」と入れることで検証できます。

画面の上へ

## こんなときは

### 動画を再生するとPCが重くなる

- 動画の再生が重い（スムーズでない）のは、PCのスペックや回線速度が関係しています。たとえば、解像度を1920x1280や1280x720にしない、フレームレートを30fps以下にする \*10、ビットレートを1Mbps以上にしない、などのことを守るだけでも動画の再生がスムーズになります。とくに解像度を大きくしすぎないことは重要です。
- また、上表を参考にしてx264guiExの設定も変更しましょう。x264のプリセットも参考になります。「x264」タブの「プリセットのロード」で「速度」を「superfast」「veryfast」「faster」「fast」のうちいずれかを選択して「GUIにロード」ボタンをクリックするとプリセットを使用できます。各プリセットの設定を比較してエンコードを繰り返し、動画再生時の負荷を抑えられる設定を見つけます。

### エンコード時間を短くしたい

- AviUtlでH.264形式の動画を出力する場合、CPUの性能が高いほうがエンコード時間を短くできます。CPUの性能が悪いと、どうしてもエンコード時間は長くなります。また、動画の再生時間を短くする、フレームレートを低くする、解像度を小さくする、シングルパスにする、などの対処法も大きな効果があります。
- さらに、x264guiExの「x264」タブにある「プリセットのロード」で、「速度」を「superfast」「veryfast」「faster」「fast」のうちいずれかを選択して「GUIにロード」ボタンをクリックすると、エンコード時間を短縮する設定になります。ただ、画質は悪くなります。エンコード時間を短くするための設定というのは、基本的に画質について妥協が必要です。

### 映像が単一の色で乱れる

- 動画再生時、設定・環境によってはピンク色のノイズで映像が乱れることがあります。この場合、対処法としては（1）ハードウェアアクセラレーションをOFFにする、（2）x264guiExの設定を変更して動画をエンコードしなおす、という方法が考えられます（詳細はニコニコ大百科を参照のこと）。
- まず（1）の方法ですが、これはPCに搭載されている動画再生支援機能をOFFにするというこ

とです。具体的には以下の作業をすることになります。

1. ニコニコ動画の再生プレイヤー上で右クリックし、「設定」を選択する。
  2. ディスプレイのアイコンをクリックする。
  3. 「ハードウェアアクセラレーションを有効化」をOFFにする。
  4. ページを更新する。
- つぎに(2)の方法ですが、これは「フレーム」タブの「参照距離」および「最大連続Bフレーム数」を小さい数値に変更して動画をエンコードすることです。とりあえず両者をそれぞれ3に設定して動画をエンコードし、様子を見てください。

[画面の上へ](#)

## その他

- ニコニコ動画に動画を投稿するさい、「エンコードしています...。」と表示されますが、これは動画のアップロード時に必ず表示されます。
- 作成したH.264動画は、[GOM PLAYER](#)や[Flavie](#)などで再生できます。後者のほうが色の確認をしっかりとできます。

[画面の上へ](#)

## 関連ページ

- このページと関連性の強いページは以下のとおりです。

ページ名	内容	重要度
<a href="#">コメント</a>	H.264で高画質についての質問など	-
<a href="#">動画の基礎知識</a>	動画作成における基本的な事項	A
<a href="#">AviUtlの使い方</a>	動画編集の方法	B+
<a href="#">AviUtl拡張編集</a>	拡張編集プラグインの使い方	B+
<a href="#">ニコエンコで変換</a>	ファイルサイズを小さくする方法	B+

- 参考になるWebサイト
  - [ニコニコ動画まとめwiki / AviUtlを使ったMP4 \(H.264\) エンコード](#)
  - [rigayaの日記兼メモ帳](#)

[画面の上へ](#)