

Important User Concepts

重要なユーザ コンセプト

目次 Table of Contents

[1. 経路制御 Trajectory Control](#)

- [1.1. 経路計画 Trajectory Planning](#)
- [1.2. 工具パスへの追従 Path Following](#)
- [1.3. 経路のプログラミング Programming the Planner](#)
- [1.4. 動作の計画 Planning Moves](#)

[2. Gコード\(NC言語\) G Code](#)

- [2.1. 初期の設定 Defaults](#)
- [2.2. 送り速度 Feed Rate](#)
- [2.3. 工具径オフセット Tool Radius Offset](#)

[3. 原点復帰 Homing](#)

[4. 工具交換 Tool Changes](#)

[5. 座標系 Coordinate Systems](#)

- [5.1. G53機械座標系 G53 Machine Coordinate](#)
- [5.2. G54-59.3ユーザ定義座標系 G54-59.3 User Coordinates](#)
- [5.3. 位置がわからなくなったら When You're Lost](#)

[6. 機械の設定 Machine Configurations](#)

このセクションではCNCをGコードで走らせる前に理解しておくべき重要なコンセプトについてひと通りカバーしています。

※この日本語版は非公式なものです。内容の完全性は保証されません。
気になる箇所はlinuxcnc.org公式のドキュメントを参照して確認してください。
(ドキュメントのフッタにオリジナルのURLがあります)

1. 経路の制御 Trajectory Control

1.1. 経路作成 Trajectory Planning

経路作成とは、Gコードプログラムで指定した工具パスをLinuxCNCが追従するという意味で、理想経路に対して、設定した装置の制限の範囲で実行されます。

Gコードのプログラムは、実際の動作では完全に再現することができません。例として、以下の1行コードのようなプログラムを記述したとします:

G1 X1 F10 (G1は直線補間移動、X1は行き先のX座標、F10は速度)

現実の世界では、移動中常にF10で示す速度で動作することはできません。なぜなら、機械は静止時から加速し、到着前に減速し停止する必要があるからです。また、あるときは指示した速度に到達するときもありますが、短い移動距離の場合は指定速度に到達する前に減速することもあります。コードのいくつかの割合では指示した速度F10で移動出来るかもしれませんが、短い移動距離の多くの移動では、指示した速度には全く届きません。「ナイーブ CAM ディテクタ/naive cam detector」という機能で「G64Pn.nnn(nには数字が入る)」指示を用いていないと、短距離の補間動作が続く場所では機械をスローダウンさせ、長距離区間では速くなる動作ムラの原因になります。

先に説明したような加減速の基本は複雑なものではありませんし、製作上の妥協でもありません。経路作成の演算はINIファイルに記述した、機械の出せる最高速度と最高加速度には必ず従います。

1.2. 工具パスへの追従 Path Following

より難解な問題としては、工具パスへの追従があげられます。Gコードプログラムが曲がり角に来たとき、経路作成の演算にはいくつかの取りうる手法がありますが、もっとも経路として正しいのは、完全に停止するまで減速し、それから次の目標点に向かって加速し始める動作です。ブレンディングという方法もあり、これはコーナーで送り速度を維持したまま通過するもので、どうしても機械の加速度に応じて角は丸くなります。ここに、トレードオフの関係が発生します:より高精度な経路追従のためにスローダウンさせることもできますし、追従度より速度維持を優先することもできます。どちらを選ぶかは、各加工に特有な、材料、工具その他の条件からGコードプログラムが妥協できる内容によって異なります。

早送り移動でも経路制御に従います。十分な距離を移動するなら低い加速度でも最高速に達しますし、経路の精度指示がなければ、正確な曲率の角ができます。

1.3. Gコードプログラミングと経路 Programming the Planner

経路制御コマンドは以下のようにになっています:

- **G61** - (イグザクトパス モード/Exact Path Mode)
Gコードでプログラムした点や経路を正確に再現しますが、そのかわり、次の座標に行くのに大きく方向転換する必要のあるときはいったん減速したり停止したりする可能性があるという事を意味します。
- **G61.1** - (イグザクトストップ モード/Exact Stop Mode)
これは各補間移動の終点ごとに確実に停止し理想的な経路を通過するかわり、走行時間の増大と加工条件の変化をまねきます。
- **G64** - (公差無視ブレンド モード/Blend Without Tolerance Mode)
「G64」は特にカスタムしていなければLinuxCNC起動時の初期設定です。G64のみでP指示のない場合はブレンドするだけで「naive cam detector」は有効になりません。「G64」と「G64 P0」はパス追従精度を犠牲にする代わりに送り速度を維持するよう経路計画させます。この機能はいくつかの工具やワーク材料でイグザクトストップが害になる場合に有効な機能で、プログラマがGコード指示よりいくらか丸まった工具パスになることを考慮に入れている限りはとてもよく動作するでしょう。G64とともにG0早送りを使用した場合は、移動時のワークとのすき間や、機械の加速度に応じた障害物を回避するのに十分な距離が確保できているかに注意が必要です。
- **G64 P- Q-** - (公差ありブレンド モードBlend With Tolerance Mode)
これにより「naive cam detector」という機能と公差つきブレンドを有効にします。仮に「G64 P0.05」とすると、連続的な送りの途中、現在の速度ではコーナー部でプログラムした経路から0.05 mm(設定した機械の動作単位によってはインチ)以上ずれそうなときは必要なだけ減速すると経路計画ユニットに伝えることになります。正確な減速量はプログラムしたコーナー部の諸元やiniファイルに記述した機械の制限によりますが、プログラマが心配すべきはただひとつ、公差だけです。公差指示による経路追従度の制限によってプログラマは完全な制御を得られます。公差ブレンドの値は必要に応じプログラムの途中で変更を記述できます。なお、「G64 P0」指示は「G64」単独での指示と同じ効果を持つことに注意してください。(これは古いGコードプログラムとの後方互換性のために必要な措置です)詳細はGコードの章の「G64P- Q-」をみてください。
 - 公差無視ブレンド *Blending without tolerance*
工具の制御点は指示された移動の間に最低1点は理想経路に接します。現在の動作(または、次の動作のブレンドが始まったあとであなたがコードの実行をポーズしたとき)の終点で完全に停止できないような速度では装置は動かされません。動作の終点からの距離は、もっともよい輪郭を保てる送りに必要なだけの大きさになります。

○ 「ナイーブ CAM ディテクタ /Naive Cam Detector」

「Q-」の値より小さな誤差となる、XYZ軸のみで構成される連続したG1直線補間動作は1つの直線に統合されます。この結合した動作は公差つきブレンドとしてもとの独立したG1動作を置き換えます。また、連続した動作のあいだ、制御点は実際の動作の終点から「P-」の値より離れて通ることはありません。制御点は各動作中最低1点は接します。動作(または、次の動作のブレンドが始まったあとであなたがコードの実行をポーズしたとき)の終点で完全に停止できないような速度では装置は動きません。直線から弧への最大のずれが「Q-」より小さいとき、G17、つまりXY平面上でのG02/G03円弧補間移動は2つの線(円弧のはじめから中点までと、中点から終点まで)に分けられます。それらの線はそのあとnaive cam アルゴリズムで処理されます。このようにして、直線-円弧、円弧-円弧、円弧-直線、それから直線-直線のそれぞれの場合ごとにnaive cam detectorの手助けを得ます。このような経路の単純化が輪郭性能を向上させます。

下図の青線は実際の機械の速度を、赤線は加速度の能力を、水平の線は計画した動作を表しています。上側のグラフでは短距離移動時に、設定した加速度の制限によって経路計画が移動速度を下げていますのがわかります。下方のグラフでは「naive cam detector」の効果で複数の動作が統合され、より計画通りの速度を保って移動しているのがわかります。

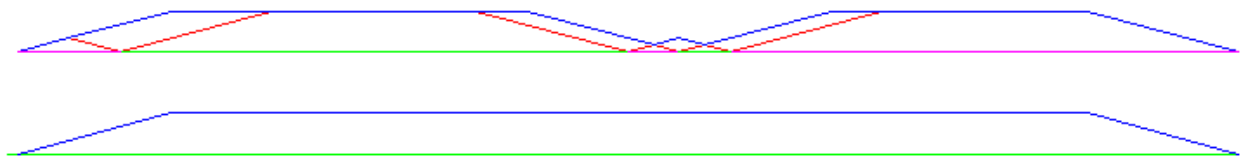


Figure 1. Naive Cam Detector

1.4. 動作の計画 Planning Moves

動作が利用する装置や材料に合うのかよく確かめてください。なぜかという、装置は現在の動作の終了時に停止しきれないような速度で動かすことはできないからです。与えられた加速度の設定値で要求されている送り速度を装置が維持できる最小距離が存在しているのです。

加減速部分ではiniファイルのMAX_ACCELERATION変数の値の半分の値を使用します。ブレードモードでは総合加速度はMAX_ACCELERATIONと同じ値まで使います。その他の場合、実際の装置の加速度はiniファイルよりはいくらか小さい値になります。

送り速度を維持するには、移動距離を速度0から送り速度まで加速し、それからまた停止するのに必要な距離より長く取っていなければなりません。MAX_ACCELERATIONの1/2の値を「**A**」、1秒あたりの送り量を「**F**」とすると、加速時間は

$$t_a = F/A$$

で、加速距離は

$$d_a = F * t_a / 2$$

で表せます。減速時間とその距離は加速時と同じですので、加減速距離は

$$d = d_a + d_a = 2 * d_a = F^2/A.$$

となります。

一例を挙げると、**F** = 1 mm、**A** = 10 mm/sec²のとき、の加減速距離は

$$1^2/10 = 1/10 = 0.1 \text{ mm}$$

となります。

そのままの加速度で送り量を5倍の毎秒 5 mmにすると、加減速距離は

$$5^2/10 = 25/10 = 2.5 \text{ mm}$$

のように大きく伸びます。

2. Gコード G Code

2.1. 初期設定 Defaults

LinuxCNCを立ち上げたとき、デフォルト値として複数のGコードやMコードを読み込みます。いまアクティブになっているコードがどれなのかはAXISユーザインタフェースのMDI(Manual Data Input/手動コード入力)モード タブの「[Active G-Codes](#)」に表示されています。これらのコードはLinuxCNCの振る舞いを定義するものですので、LinuxCNCを運用する前にそれぞれのコードが何を意味しているのか理解しておくことが重要です。デフォルトの読み込みコードはLinuxCNCが起動したあとで設定の書いてあるGコードファイルを実行すると変更することが可能です。一番良いのは、起動時の初期設定が変化しないと思わず、Gコードのファイルの先頭でその作業に必要な設定をセットすることです。Gコードクイックリファレンスのページを印刷しておくでG-Code [Quick Reference](#) それぞれのコードの意味を忘れてしまったときの助けになります。

2.2. 送り速度 Feed Rate

送り速度の値をどうセットするかは動作指令に回転を含むかどうかにより変化します。装置に回転軸があったり旋盤の場合は [Feed Rate](#) セクションをよく読んで理解してください。

2.3. 工具径オフセット Tool Radius Offset

工具径オフセット (G41/42)によって対象にえぐりこむことなく工具を輪郭に沿って動作させることができます。現在取り付けである工具径から不可能なオフセット指示は加工誤差になります。オフセット指示との実際の工具径の誤差の小さい工具ほど、同じパスを走らせたときの誤差が小さくなります。これは、(粗加工として)誤差のない工具より外側を通る経路をプログラムで指示できることを意味します。詳細な情報は [Cutter Compensation](#) セクションをみてください。

3. 原点復帰 Homing

LinuxCNCをスタートしたら、プログラムを実行したりMDIでコマンドを打つよりも先に、まず各軸を原点復帰させなければなりません。装置に原点センサがないときは各軸ごとに合いマークを使うことで毎回ほぼ同じところに手動で座標を持っていくことができます。原点復帰が完了すると、INIファイルで設定したソフトウェアリミットが使えるようになります。

原点復帰の強制を解除したり、最小限の構成で済ませたいのなら、iniファイルの `[TRAJ]` セクションに「`NO_FORCE_HOMING = 1`」と書き加えることで実現できます。原点復帰に関する詳細な解説は Integrator Manual に書かれています。

4. 工具交換 Tool Changes

手動工具交換にはいくつかのオプションがあります。Integrator Manualの `[EMCIO]` セクションにこれらオプション設定についての情報があります。また、User Manualの `G28`と`G30`についてのセクションも参考になるでしょう。

5. 座標系 Coordinate Systems

座標系のシステムはCNCを使うとき真っ先に混乱するところです。CNCマシンを動かす前に、LinuxCNCで使われている座標系の基礎を押さえておきましょう。LinuxCNCの座標系についてもっと深い話は[Coordinate System](#) セクションに書いてあります。

5.1. G53機械座標系 Machine Coordinate

原点を取った直後、G53命令で機械座標系にセットすると、原点を取った軸の値は「0」になっているはずです。

- 注:ほかの座標系や工具オフセットは原点復帰前には変更できません。

機械座標系で動作させたいときにだけ、機械座標系で座標指示したプログラムを書きます。

5.2. G54-59.3ユーザ座標系 User Coordinates

普通は運用時G54ユーザ座標を使うことが多いと思います。ユーザ座標系を適用しAXISの設定が *Position: Relative Actual* なら機械原点に来たときにDROに線の入った小さな青丸が表示されます。一時的に使った座標系オフセットをリセットしたいときには、装置のメニューから設定するか、Gコードファイルの末尾に「*G10 L2 P1 X0 Y0 Z0*」をプログラムしておいてください。「P」の後ろの数字は、クリアしたいユーザ座標系オフセットが入っている格納場所の番号に合わせてください。

- ユーザ座標系に格納したオフセット座標はLinuxCNCを終了したときにも保持されます。
- 軸の「*Touch Off*」ボタンをつかうと、選択されているユーザ座標系にオフセットを掛けます。

3. 迷子になったら When You're Lost

ここが原点のばすだと思っている場所でDROが「0,0,0」を指さなくなってしまったときは、おそらくいくつかのオフセット命令がGコードプログラムによって設定されているためでしょうから、これらを解除してあげる必要があります。

- 「*G53 G0 X0 Y0 Z0*」で機械座標系の原点に戻ります
- 「*G92.1*」コマンドですべてのG92オフセットがクリアされます
- 「*G54*」でG54ユーザ座標系に移行します
- 「*G54*」命令で座標系をセットするのと、機械座標系でオフセット指定座標系選択命令「*G10 L2 P1 X0 Y0 Z0 R0*」とするのは同じ効果です
- 「*G49*」で工具オフセットをキャンセルします
- DROをユーザ座標系表示にするには「menu」から「Relative Coordinate Display」を選択します

これで機械原点「X0Y0Z0」地点にいて、ユーザ座標系と機械座標系が一致した状態になりました。

6. 装置の設定 Machine Configurations

以下の図は典型的なフライス盤の工具軸やテーブルの移動方向と、それらのリミットスイッチの配置を表したものです。テーブルが移動するときと、工具が移動するときでデカルト座標の正の向きが逆転するので注意してください。この進行方向の逆転によって材料からみた工具の動作が正しく統一されます。

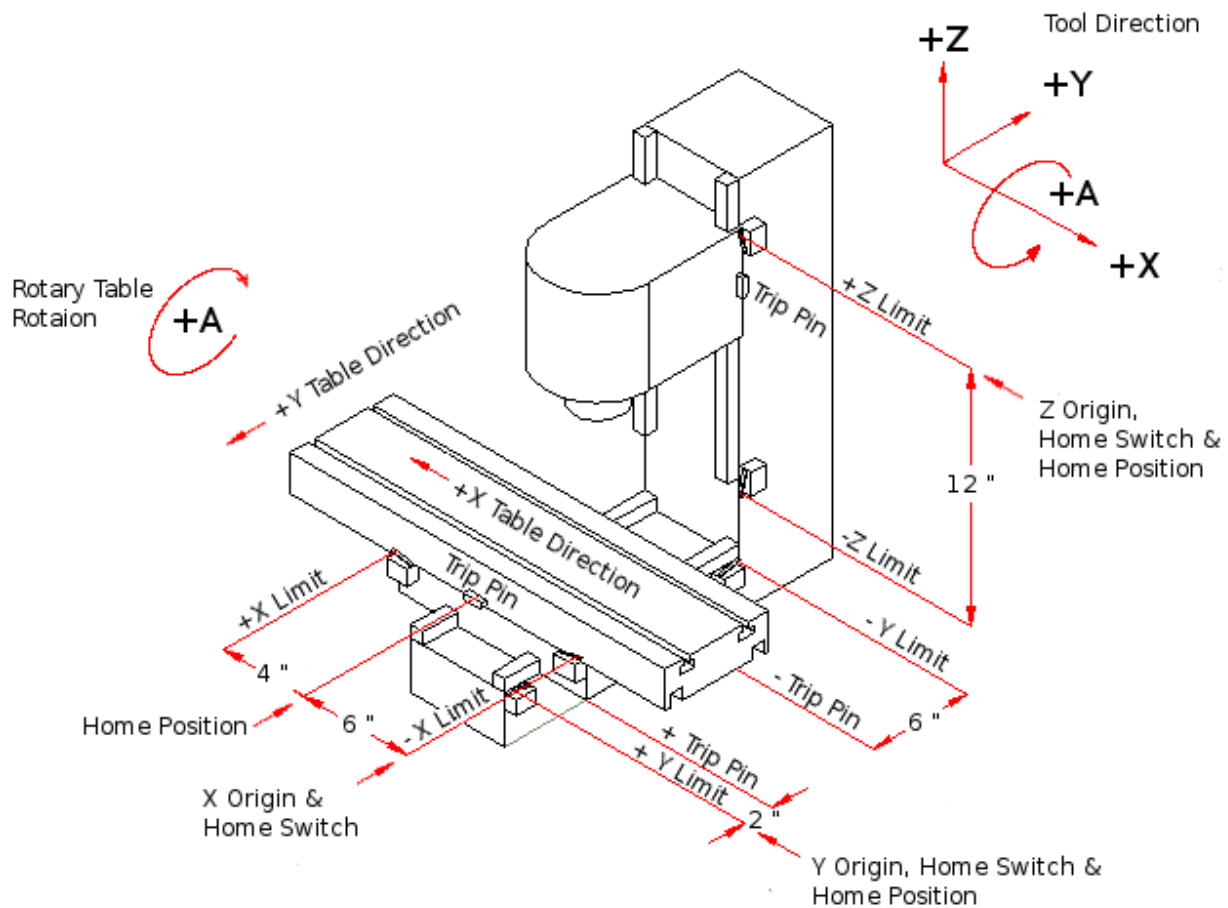


Figure 2. フライスでの軸方向の設定 Mill Configuration

下の図では典型的な旋盤での工具の移動方向やリミットの配置が示されています。

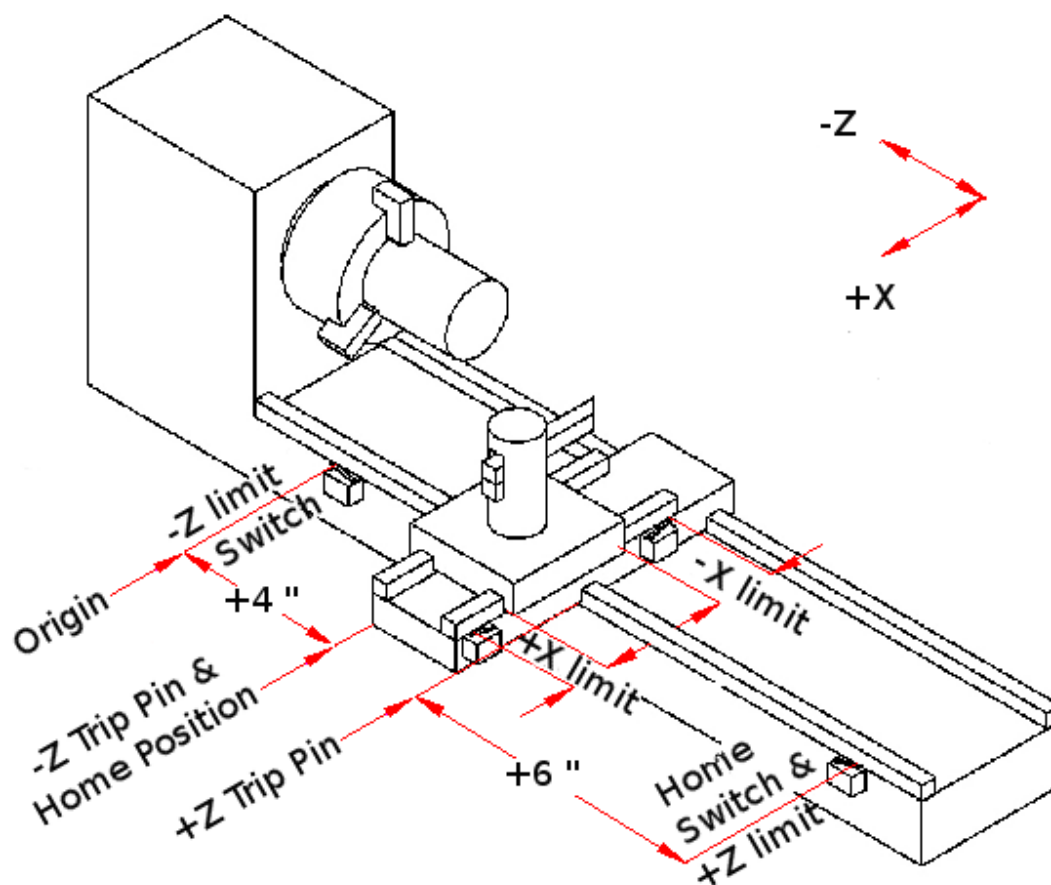


Figure 3. 旋盤での軸方向の設定 Lathe Configuration

Last updated 2013-11-28 23:49:20 MST

日本語版最終更新:2014-04-08 JST