# Manuel de l’utilisateur V2.7.5, 2016-07-13

# Table des matières

**1 Avant-propos** **1**

**2 LinuxCNC** **3**

2.1 Introduction . . . 3
2.2 Comment fonctionne LinuxCNC . . . 3
2.3 Interfaces utilisateur graphiques . . . 4
2.4 Panneaux de contrôle virtuels . . . 10
2.5 Langues . . . 10
2.6 Penser comme un opérateur sur CNC . . . 10
2.7 Modes opératoires . . . 11

**I Les interfaces utilisateur** **12**

**3 L'interface graphique AXIS** **13**

3.1 Introduction . . . 13
3.2 Commencer avec AXIS . . . 14
3.2.1 Une session typique avec AXIS . . . 14
3.3 Eléments de la fenêtre d'AXIS . . . 15
3.3.1 Options des menus . . . 15
3.3.2 Boutons de la barre d'outils . . . 18
3.3.3 Zones d'affichage graphique du programme . . . 18
3.3.4 Zone texte d'affichage du programme . . . 20
3.3.5 Contrôle manuel . . . 20
3.3.6 Données manuelles (MDI) . . . 22
3.3.7 Correcteurs de vitesse . . . 23
3.3.8 Correcteur de vitesse de broche . . . 23
3.3.9 Vitesse de Jog . . . 23
3.3.10 Vitesse Maxi . . . 23
3.4 Raccourcis clavier . . . 23
3.5 Affichage de l'état de LinuxCNC (LinuxCNCtop) . . . 24

3.6 Entrée de données en texte (MDI) . . . 25
3.7 axis-remote: Commande à distance de l'interface graphique d'AXIS . . . 26
3.8 hal_manualtoolchange: Dialogue de changement d'outil manuel . . . 26
3.9 Modules en Python . . . 26
3.10 Utiliser AXIS en mode tour . . . 26
3.11 Configuration avancée d'AXIS . . . 27
3.11.1 Filtres de programme . . . 27
3.11.2 La base de données des ressources X . . . 28
3.11.3 Manivelle de jog . . . 29
3.11.4 ~/.axisrc . . . 29
3.11.5 Éditeur externe . . . 29
3.11.6 Panneau de contrôle virtuel . . . 29
3.11.7 Commentaires spéciaux . . . 29

**4 L'utilitaire graphique NGCGUI 31**
4.1 Vue d'ensemble . . . 31
4.2 Configurations fournies en exemple. . . . 32
4.3 Librairies . . . 32
4.4 Intégration de ngcgui dans Axis . . . 33
4.4.1 Traceur Truetype . . . 34
4.4.2 Exemples d'INI . . . 35
4.5 Besoins des sous-programmes . . . 37
4.6 Exemple, découpe pour DB25 . . . 39
4.7 Création d'un sous-programme . . . 42

**5 L'interface graphique Touchy 43**
5.1 Panneau de Configuration . . . 43
5.1.1 Connections avec HAL . . . 43
5.1.1.1 Contrôles requis . . . 44
5.1.1.2 Contrôles optionnels . . . 44
5.1.1.3 Voyants de panneau optionnels . . . 44
5.1.2 Recommandé pour toutes les configurations . . . 44
5.2 Réglages . . . 44
5.2.1 Macros . . . 44

**6 L'interface graphique TkLinuxCNC** **46**
6.1 Introduction . . . 46
6.2 Utiliser TkLinuxCNC . . . 47
6.2.1 Une session typique avec TkLinuxCNC . . . 47
6.3 Éléments affichés par TkLinuxCNC . . . 48
6.3.1 Boutons principaux . . . 48
6.3.2 Barre de statut des différents offsets . . . 48
6.3.3 Zone d'affichage des coordonnées . . . 48
6.3.3.1 Parcours d'outil . . . 49
6.3.4 Contrôle en automatique . . . 49
6.3.4.1 Boutons de contrôle . . . 49
6.3.4.2 Zone texte d'affichage du programme . . . 49
6.3.5 Contrôle en manuel . . . 49
6.3.5.1 Touches implicites . . . 49
6.3.5.2 Le groupe de boutons *Broche* . . . 50
6.3.5.3 Le groupe de boutons *Arrosage* . . . 50
6.3.6 Entrée manuelle de G-code (MDI) . . . 50
6.3.6.1 MDI: . . . 51
6.3.6.2 G-Codes actifs . . . 51
6.3.7 Vitesse de Jog . . . 51
6.3.8 Correcteur de vitesse d'avance travail . . . 51
6.3.9 Correcteur de vitesse de broche . . . 51
6.4 Raccourcis clavier . . . 51

**7 L'interface graphique MINI** **53**
7.1 Introduction . . . 53
7.2 L'écran . . . 54
7.3 Barre de menu . . . 54
7.4 Barre de boutons de contrôle . . . 56
7.4.1 MANUEL . . . 56
7.4.2 AUTO . . . 57
7.4.3 MDI . . . 58
7.4.4 VITESSE-0 et CONTINUER . . . 58
7.4.5 ABANDON . . . 58
7.4.6 Arrêt d'Urgence . . . 58
7.5 Colonne de gauche . . . 59
7.5.1 Afficheurs de position des axes . . . 59
7.5.2 Correcteur de vitesse travail . . . 59
7.5.3 Messages . . . 59

7.6 Colonne de droite . . . 59
7.6.1 Editeur de texte . . . 60
7.6.2 Parcours d'outil . . . 61
7.6.3 Page des outils . . . 62
7.6.4 Page des offsets . . . 62
7.7 Raccourcis clavier . . . 63
7.7.1 Touches courantes . . . 63
7.7.2 Mode Manuel . . . 64
7.7.3 Mode Auto . . . 65
7.8 Divers . . . 65

**8 L'interface textuelle KEYSTICK 66**
8.1 Introduction . . . 66
8.2 Installation . . . 67
8.3 Utilisation . . . 67

**II L'utilisation de LinuxCNC 68**

**9 Concepts importants pour l'utilisateur 69**
9.1 La configuration machine . . . 69
9.2 Contrôle de trajectoire . . . 71
9.2.1 La planification de trajectoire . . . 71
9.2.2 Le suivi du parcours . . . 72
9.2.3 La programmation du planificateur . . . 72
9.2.4 Planification des mouvements . . . 73
9.3 G-code . . . 73
9.3.1 Par défaut . . . 73
9.4 Vitesse d'avance . . . 73
9.5 Compensation de rayon d'outil . . . 74
9.6 Prise d'origine machine . . . 74
9.7 Changement d'outil . . . 74
9.8 Systèmes de coordonnées . . . 74
9.8.1 G53 Coordonnées machine . . . 74
9.8.2 G54 à 59.3 Coordonnées utilisateur . . . 74
9.8.3 Quand vous êtes perdu . . . 75

**10 Aperçu global d'une machine CNC** **76**
10.1 Composants mécaniques . . . 76
10.1.1 Axes . . . 76
10.1.1.1 Axes linéaires primaires . . . 76
10.1.1.2 Axes linéaires secondaires . . . 76
10.1.1.3 Axes rotatifs . . . 76
10.1.2 Broche . . . 77
10.1.3 Arrosages . . . 77
10.1.4 Correcteurs de vitesse d'avance et de broche . . . 77
10.1.5 Bouton d'effacement de bloc . . . 77
10.1.6 Bouton d'arrêt optionnel du programme . . . 77
10.2 Composants de contrôle et de données . . . 77
10.2.1 Axes linéaires . . . 77
10.2.2 Axes linéaires secondaires . . . 77
10.2.3 Axes rotatifs . . . 77
10.2.4 Point contrôlé . . . 77
10.2.5 Mouvement linéaire coordonné . . . 78
10.2.6 Vitesse d'avance . . . 78
10.2.7 Arrosage . . . 78
10.2.8 Temporisation . . . 78
10.2.9 Unités . . . 78
10.2.10 Position courante . . . 78
10.2.11 Choix du plan de travail . . . 79
10.2.12 carrousel d'outils . . . 79
10.2.13 Changeur d'outil . . . 79
10.2.14 Chargeur de pièce . . . 79
10.2.15 Chargeur de pièces . . . 79
10.2.16 Boutons des correcteurs de vitesses . . . 79
10.2.17 Modes de contrôle de trajectoire . . . 79
10.3 Interaction de l'interpréteur avec les boutons . . . 79
10.3.1 Boutons de correction de vitesses . . . 79
10.3.2 Bouton d'effacement de bloc . . . 80
10.3.3 Bouton d'arrêt optionnel du programme . . . 80

**11 Systèmes de coordonnées et décalages** **82**
11.1 Introduction . . . 82
11.2 Commande en coordonnées machine: G53 . . . 82
11.3 Décalages pièce (G54 à G59.3) . . . 83
11.3.1 Système de coordonnées par défaut . . . 84
11.3.2 Réglage des décalages avec G10 . . . 84
11.4 Décalages d'axes G92 . . . 84
11.4.1 Les commandes G92 . . . 85
11.4.2 Réglage des valeurs de G92 . . . 85
11.4.3 Précautions avec G92 . . . 85
11.5 Exemple de programme utilisant les décalages d'axes . . . 86

**12 Les compensations d'outil** **88**
12.1 Compensation de longueur d'outil . . . 88
12.1.1 Toucher . . . 88
12.1.2 Utilisation de G10 L1/L10/L11 . . . 89
12.2 Table d'outils . . . 89
12.2.1 Format de la table d'outils . . . 89

**13 Fichier d'outils et compensations** **91**
13.1 Fichier d'outils . . . 91
13.2 Compensation d'outil . . . 91
13.3 Compensation de longueur d'outil . . . 91
13.4 Compensation de rayon d'outil . . . 92
13.4.1 Vue générale . . . 93
13.4.1.1 Table d'outils . . . 93
13.4.1.2 Programmation des mouvements d'entrée . . . 93
13.4.1.3 Mouvement en Z . . . 94
13.4.1.4 Mouvement en vitesse rapide . . . 94
13.4.1.5 Bonne pratique . . . 94
13.4.2 Exemples de profils . . . 95
13.4.2.1 Profil extérieur . . . 95
13.4.2.2 Profil intérieur . . . 96
13.5 Deux exposés sur les compensations d'outil . . . 96
13.5.1 Compensation de rayon d'outil, détails . . . 96
13.5.1.1 Instructions de programmation . . . 97
13.5.1.2 La valeur de D . . . 97
13.5.1.3 Table d'outils . . . 98
13.5.1.4 Deux types de contour . . . 98

13.5.1.5 Contour sur le profil du matériau . . . 98
13.5.1.6 Contour sur le parcours d'outil . . . 98
13.5.1.7 Erreurs de programmation et limitations . . . 99
13.5.2 Premier mouvement . . . 100
13.5.2.1 Programmation des mouvements d'entrée . . . 101
13.5.2.2 Méthode générale . . . 101
13.5.2.3 Méthode simple . . . 102
13.5.2.4 Autres points où est exécutée la compensation de rayon d'outil . . . 103
13.5.2.5 Algorithmes pour compensation de rayon d'outil . . . 103
13.5.3 Exemples de Jon Elson . . . 103

**14 Vue générale du langage G-codes de LinuxCNC 106**
14.1 Brève description du G-code de LinuxCNC . . . 106
14.2 Format des paramètres du G-code . . . 106
14.3 Format d'une ligne . . . 106
14.4 Caractère d'effacement de bloc . . . 107
14.5 Numéro de ligne . . . 107
14.6 Les mots . . . 107
14.7 Les nombres . . . 108
14.8 Paramètres (Variables) . . . 108
14.9 Paramètres numérotés . . . 109
14.10 Paramètres de sous-programme . . . 110
14.11 Paramètres nommés . . . 111
14.12 Paramètres nommés prédéfinis . . . 111
14.13 Paramètres système . . . 113
14.14 Expressions . . . 113
14.15 Opérateurs binaires . . . 113
14.16 Fonctions . . . 114
14.17 Répétitions d'items . . . 115
14.18 Ordre des items . . . 115
14.19 Commandes et modes machine . . . 115
14.20 Coordonnées polaires . . . 115
14.21 Groupes modaux . . . 117
14.22 Commentaires . . . 118
14.23 Messages . . . 119
14.24 Enregistrement des mesures . . . 119
14.25 Log général . . . 119
14.26 Messages de débogage . . . 119
14.27 Paramètres dans les commentaires . . . 119

14.28 Exigences des fichiers . . . 120
14.29 Taille des fichiers . . . 120
14.30 Ordre d'exécution . . . 120
14.31 G-Code: Bonnes pratiques . . . 121
14.31.1 Utiliser un nombre de décimales approprié . . . 121
14.31.2 Utiliser les espaces de façon cohérente . . . 121
14.31.3 Préférer le *format centre* pour les arcs . . . 121
14.31.4 Placer les codes modaux importants au début des programmes . . . 121
14.31.5 Ne pas mettre trop de choses sur une ligne . . . 121
14.31.6 Ne pas régler et utiliser un paramètre sur la même ligne . . . 121
14.31.7 Ne pas utiliser les numéros de ligne . . . 122
14.31.8 Lorsque plusieurs systèmes de coordonnées sont déplacés . . . 122
14.32 Axes rotatifs et linéaires . . . 122
14.33 Messages d'erreur courants . . . 122

**15 Tout le G-code de LinuxCNC** **123**
15.1 Conventions d'écriture du G-code . . . 123
15.2 Table d'index du G-code . . . 123
15.3 G0 Interpolation linéaire en vitesse rapide . . . 124
15.4 G1 Interpolation linéaire en vitesse travail . . . 125
15.5 G2, G3 Interpolation circulaire en vitesse travail . . . 125
15.5.1 Arc au format centre (format recommandé) . . . 126
15.5.2 Exemples d'arcs au format centre . . . 127
15.5.3 Arcs au format rayon (format non recommandé) . . . 129
15.6 G4 Tempo . . . 130
15.7 G5 Spline cubique . . . 130
15.8 G5.1 Spline quadratique . . . 131
15.9 G5.2 G5.3 Block NURBS . . . 131
15.10 G7 Mode diamètre sur les tours . . . 133
15.11 G8 Mode rayon sur les tours . . . 133
15.12 G10 L1 Ajustements dans la table d'outils . . . 133
15.13 G10 L2 Établissement de l'origine d'un système de coordonnées . . . 133
15.14 G10 L10 modifie les offsets d'outil dans la table d'outils . . . 134
15.15 G10 L11 modifie les offsets d'outil dans la table d'outils . . . 135
15.16 G10 L20 Établissement de l'origine d'un système de coordonnées . . . 135
15.17 G17 à G19.1 Choix du plan de travail . . . 136
15.18 G20, G21 Choix des unités machine . . . 136
15.19 G28, G28.1 Aller à une position prédéfinie . . . 136
15.20 G30, G30.1 Aller à une position prédéfinie . . . 136

15.21 G33 Mouvement avec broche synchronisée . . . 137
15.22 G33.1 Taraudage Rigide . . . 138
15.23 G38.x Mesure au palpeur . . . 139
15.24 G40 Révocation de la compensation de rayon d'outil . . . 140
15.25 G41, G42 Compensation de rayon d'outil . . . 140
15.26 G41.1, G42.1 Compensation dynamique d'outil . . . 140
15.27 G43 Activation de la compensation de longueur d'outil . . . 141
15.28 G43.1 Compensation dynamique de longueur d'outil . . . 141
15.29 G49 Révocation de la compensation de longueur d'outil . . . 141
15.30 G53 Mouvement en coordonnées absolues . . . 142
15.31 G54 à G59.3 Choix du système de coordonnées . . . 142
15.32 G61, G61.1 Contrôle de trajectoire exacte . . . 143
15.33 G64 Contrôle de trajectoire continue avec tolérance . . . 143
15.34 G73 Cycle de perçage avec brise copeaux . . . 143
15.35 G76 Cycle de filetage préprogrammé . . . 144
15.36 Les cycles de perçage G81 à G89 . . . 146
15.36.1 Mots communs . . . 146
15.36.2 Mots *sticky* . . . 146
15.36.3 Répétition de cycle . . . 147
15.36.4 Mode de retrait . . . 147
15.36.5 Erreurs des cycles de perçage . . . 147
15.36.6 Mouvement préliminaire et Intermédiaire . . . 147
15.36.7 Pourquoi utiliser les cycles de perçage? . . . 147
15.37 G80 Révocation des codes modaux . . . 148
15.38 G81 Cycle de perçage . . . 149
15.39 G82 Cycle de perçage avec temporisation . . . 152
15.40 G83 Cycle de perçage avec débourrage . . . 152
15.41 G84 Cycle de taraudage à droite . . . 153
15.42 G85 Cycle d'alésage, sans temporisation, retrait en vitesse travail . . . 153
15.43 G86 Cycle d'alésage, arrêt de broche, retrait en vitesse rapide . . . 153
15.44 G87 Alésage inverse . . . 153
15.45 G88 Alésage, arrêt de broche, retrait en manuel . . . 153
15.46 G89 Cycle d'alésage, temporisation, retrait en vitesse travail . . . 154
15.46.1 Pourquoi utiliser les cycles de perçage ? . . . 154
15.47 G90, G91: Modes de déplacement . . . 155
15.48 G90.1, G91.1: Mode de déplacement en arc (I, J et K) . . . 156
15.49 G92 Décalage d'origine des systèmes de coordonnées . . . 156
15.50 G92.1, G92.2 Remise à zéro des décalages des systèmes de coordonnées . . . 156
15.51 G92.3 Restauration des décalages d'axe . . . 156
15.52 G93, G94, G95: Choix des modes de vitesse . . . 157
15.53 G96, G97: Modes de contrôle de la broche . . . 157
15.54 G98, G99: Options du plan de retrait . . . 157

**16 Les M-codes** **159**
16.1 Table des M-codes . . . 159
16.2 M0, M1, pause dans le programme . . . 159
16.3 M2, M30, fin de programme . . . 160
16.4 M60, pause pour déchargement pièce . . . 160
16.5 M3, M4, M5 Contrôle de la broche . . . 160
16.6 M6 Appel d'outil . . . 160
16.6.1 Changement d'outil manuel . . . 160
16.6.2 Changement d'outil . . . 161
16.7 M7, M8, M9 Contrôle de l'arrosage . . . 161
16.8 M19 Orientation de la broche . . . 161
16.9 M48, M49 Contrôle des correcteurs de vitesse . . . 162
16.10 M50 Contrôle du correcteur de vitesse travail . . . 162
16.11 M51 Contrôle du correcteur de vitesse broche . . . 162
16.12 M52 Contrôle de vitesse adaptative . . . 162
16.13 M53 Contrôle de la coupure de vitesse . . . 162
16.14 M61 Correction du numéro de l'outil courant . . . 163
16.15 M62 à M65 Contrôle de bits de sortie numérique . . . 163
16.16 M66 Contrôle d'entrée numérique et analogique . . . 163
16.17 M67 Contrôle de sortie analogique . . . 164
16.18 M68 Contrôle de sortie analogique directe . . . 164
16.19 M70 Enregistrement de l'état modal . . . 165
16.20 M71 Invalidation de l'état modal enregistré . . . 165
16.21 M72 Restauration de l'état modal . . . 165
16.22 M73 Enregistrement et auto-restauration de l'état modal . . . 166
16.23 Restauration sélective de l'état modal par le test de paramètres prédéfinis . . . 167
16.24 M100 à M199 Commandes définies par l'utilisateur . . . 167

**17 Les O-codes** **169**
17.1 Utilisation des O-codes . . . 169
17.2 Sous-programmes: **sub**, **endsub**, **return**, **call** . . . 169
17.3 Boucles: **do**, **while**, **endwhile**, **break**, **continue** . . . 170
17.4 Conditionnel: **if**, **elseif**, **else**, **endif** . . . 171
17.5 Répétition: **Repeat** . . . 171
17.6 Indirection . . . 172
17.7 Appel de fichier . . . 172

**18 Les autres codes** **173**
18.1 F: Réglage de la vitesse d'avance travail . . . 173
18.2 S: Réglage de la vitesse de rotation de la broche . . . 173
18.3 T: Choix de l'outil . . . 173

**19 Exemples de fichiers G-Code** **175**
19.1 Exemples pour une fraiseuse . . . 175
19.1.1 Fraisage hélicoïdal d'un orifice . . . 175
19.1.2 Rainurage . . . 175
19.1.3 Palpage d'une grille rectangulaire de points . . . 176
19.1.4 Amélioration du palpage d'une grille rectangulaire de points . . . 177
19.1.5 Mesure de longueur d'outil . . . 178
19.1.6 Mesure d'un alésage au palpeur . . . 178
19.1.7 Compensation de rayon d'outil . . . 178
19.2 Exemples pour un tour . . . 179
19.2.1 Filetage . . . 179

**20 Particularités des tours** **180**
20.1 Mode tour . . . 180
20.2 Fichier d'outils . . . 180
20.3 Orientations des outils de tournage . . . 180
20.4 Correction d'outil . . . 184
20.4.1 Offset d'outil en X . . . 184
20.4.2 Séquence typique de correction d'outil en X: . . . 184
20.4.3 Offset d'outil en Z . . . 184
20.4.4 Séquence typique de correction d'outil en Z: . . . 184
20.4.5 Machine avec tous les outils compensés . . . 185
20.4.6 Séquence typique de décalage du système de coordonnées: . . . 185
20.5 Mouvements avec broche synchronisée . . . 185
20.6 Arcs . . . 185
20.6.1 Les arcs et la cinématique du tour . . . 185
20.6.2 Mode rayon et mode diamètre . . . 186
20.7 Parcours d'outil . . . 186
20.7.1 Point contrôlé . . . 186
20.7.2 Tourner les angles sans compensation d'outil . . . 186
20.7.3 Tournage avec rayon extérieur . . . 188
20.7.4 Utiliser la compensation d'outil . . . 190

**21 Différences avec RS274/NGC** **192**
21.1 Changements entre RS274/NGC et LinuxCNC . . . 192
21.1.1 Position après un changement d'outil . . . 192
21.1.2 Les paramètres de décalage sont dans l'unité du fichier ini . . . 192
21.1.3 Table d'outils, longueur et diamètre sont dans l'unité du fichier ini . . . 192
21.1.4 G84, G87 ne sont pas implémentés . . . 192

21.1.5 G28, G30 avec des mots d'axe . . . 192
21.2 Ajouts à RS274/NGC . . . 193
21.2.1 Codes de filetage G33 et G76 . . . 193
21.2.2 G38.2 . . . 193
21.2.3 G38.3 à G38.5 . . . 193
21.2.4 Les O-codes . . . 193
21.2.5 M50 à M53 Correcteurs de vitesse . . . 193
21.2.6 M61 à M66 . . . 193
21.2.7 G43, G43.1 . . . 193
21.2.8 G41.1, G42.1 Compensation dynamique . . . 193
21.2.9 G43 sans le mot H . . . 193
21.2.10 U, V et W axes . . . 194

**22 Image-to-gcode: Usiner un depth maps** **195**
22.1 Qu'est-ce qu'un *depth map*? . . . 195
22.2 Intégrer image-to-gcode dans l'interface utilisateur d'AXIS . . . 195
22.3 Utilisation d'image-to-gcode . . . 196
22.4 Les différentes options . . . 196
22.4.1 Unités . . . 196
22.4.2 Invert Image . . . 196
22.4.3 Normalize Image . . . 196
22.4.4 Expand Image Border . . . 196
22.4.5 Tolerance (unités) . . . 196
22.4.6 Pixel Size (unités) . . . 196
22.4.7 Plunge Feed Rate (unités par minute) . . . 196
22.4.8 Feed Rate (unités par minute) . . . 197
22.4.9 Spindle Speed (RPM) . . . 197
22.4.10 Scan Pattern . . . 197
22.4.11 Scan Direction . . . 197
22.4.12 Depth (unités) . . . 197
22.4.13 Step Over (pixels) . . . 197
22.4.14 Tool Diameter . . . 197
22.4.15 Safety Height . . . 197
22.4.16 Tool Type . . . 197
22.4.17 Lace bounding . . . 198
22.4.18 Contact angle . . . 198
22.4.19 Offset d'ébauche et profondeur par passe d'ébauche . . . 198

**23 Glossary** **199**

**24 Legal Section** **204**

24.1 Copyright Terms . . . . . 204

24.2 GNU Free Documentation License . . . . . 204

**25 Index** **208**

The LinuxCNC Team

Ce manuel est en évolution permanente. Si vous voulez nous aider à son écriture, sa rédaction, sa traduction ou la préparation des graphiques, merci de contactez n'importe quel membre de l'équipe de traduction ou envoyez un courrier électronique à emc-users@lists.sourceforge.net.

Copyright © 2000-2012 LinuxCNC.org

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".. If you do not find the license you may order a copy from Free Software Foundation, Inc. 59 Temple Place, Suite 330 Boston, MA 02111-1307

LINUX® is the registered trademark of Linus Torvalds in the U.S. and other countries. The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.

Permission est donnée de copier, distribuer et/ou modifier ce document selon les termes de la « GNU Free Documentation License », Version 1.3 ou toute version ultérieure publiée par la « Free Software Foundation »; sans sections inaltérables, sans texte de couverture ni quatrième de couverture. Une copie de la licence est incluse dans la section intitulée « GNU Free Documentation License ». Si vous ne trouvez pas la licence vous pouvez en commander un exemplaire chez Free Software Foundation, Inc. 59 Temple Place, Suite 330 Boston, MA 02111-1307

(La version de langue anglaise fait foi)

**AVIS**

La version Française de la documentation de LinuxCNC est toujours en retard sur l'originale faute de disponibilité des traducteurs.
Il est recommandé d'utiliser la documentation en Anglais chaque fois que possible.
Si vous souhaitez être un traducteur bénévole pour la documentation française de LinuxCNC, merci de nous contactez.

**NOTICE**

The French version of the LinuxCNC documentation is always behind the original fault availability of translators.
It's recommended to use the English documentation whenever possible.
If you would like to be a volunteer editor for the French translation of LinuxCNC, please contact us.

# Chapitre 1

# Avant-propos

LinuxCNC est souple et modulaire. Ces attributs l'ont fait apparaître à certains comme un brouillon de petits morceaux confus, ils se sont demandé pourquoi il en était ainsi. Cette page tentent de répondre à cette question avant que vous lecteurs, ne plongiez dedans pour vous faire votre propre idée.

EMC a débuté à l'institut national des standards et des technologies des Etats Unis, le NIST. Il a mûri comme un logiciel fonctionnant sur le système d'exploitation Unix. Unix le rendait différent. Très tôt des développeurs Unix ont apporté une série d'idées concernant l'écriture du code, c'est devenu une écriture selon «la tradition d'Unix». Les premiers auteurs de LinuxCNC ont suivi cette voie.

Eric S. Raymond, dans son livre The Art of Unix Programming, résume la philosophie Unix par la philosophie largement utilisée en ingéniérie, le principe KISS Keep it Simple, Stupid Reste Simple, Crétin ou Sois Simple et Concis. Puis il décrit sa vision selon laquelle cette philosophie globale s'applique en tant que norme culturelle Unix, bien qu'on trouve sans surprise de graves entorses à la plupart des règles Unix suivantes:
– Règle de modularité: Ecrire des éléments simples reliés par de bonnes interfaces.
– Règle de clarté: La Clarté vaut mieux que l'ingéniosité.
– Règle de composition: Concevoir des programmes qui peuvent être reliés à d'autres programmes.
– Règle de séparation: Séparer les règles du fonctionnement; Séparer les interfaces du mécanisme. [1]
Monsieur Raymond offre d'autres règles mais ces quatre là décrivent les caractéristiques essentielles du système de contrôle de mouvement LinuxCNC.

La règle de *Modularité* est critique. Tout au long de ces manuels, vous trouverez des discussions à propos de l'interpréteur ou à propos des planificateurs de tâche ou de mouvement ou encore à propos de HAL. Chacun d'eux est un module ou un ensemble de modules. Cette modularité vous permettra de ne connecter entre elles que les parties dont vous avez besoin pour le bon fonctionnement de votre machine.

La règle de *clarté* est essentielle. LinuxCNC est en perpétuelle évolution, il n'est pas terminé et ne le sera jamais. Il est assez complet pour piloter toutes les machines que nous avons voulu qu'il pilote. Une bonne partie de cette évolution est atteinte parce que les utilisateurs et les développeurs peuvent voir le travail des autres et construire sur ce qui est déjà fait.

La règle de *composition* nous permet de concevoir et de construire un contrôleur à partir des nombreux modules existants, en les rendant connectables entre eux. Nous obtenons cette connectivité en appliquant une interface standard à tous les modules et en suivant ce standard.

La règle de *séparation* exige que chaque petite chose soit faite par une partie distincte. En séparant les fonctions, le dépannage est rendu plus aisé, le remplacement de modules par d'autres peut être fait à l'intérieur du système et la comparaison s'effectuer facilement.

Qu'apporte la fameuse «tradition d'Unix» à vous, utilisateurs de LinuxCNC. Elle signifie que vous pourrez faire des choix sur la façon d'utiliser le système. Beaucoup de ces choix affecteront les parties intégrées à la machine, mais beaucoup également affecteront la manière dont vous utiliserez votre machine. Au cours de votre lecture, vous trouverez différents endroits où vous pourrez faire des comparaisons. Finalement vous pourrez dire «J'utiliserai cette interface plutôt que telle autre» ou, «J'écrirai cette nouvelle partie de telle manière plutôt que de telle autre.» Tout au long de ces manuels nous décrirons l'étendue des possibilités de LinuxCNC actuellement disponibles.

1. Trouvé sur http://fr.wikipedia.org/wiki/Philosophie_d%27Unix, 09/09/2008

Puisque vous commencez votre voyage dans l'utilisation de LinuxCNC nous vous proposons ces deux citations de précaution.

- Pour paraphraser les paroles de Doug Gwyn sur UNIX: "LinuxCNC n'a pas été conçu pour empêcher ses utilisateurs de commettre des actes stupides, car cela les empêcherait aussi de réaliser des actes ingénieux."
- De même les paroles de Steven King: "LinuxCNC est convivial. Cependant Unix ne précise pas vraiment avec qui."

# Chapitre 2

# LinuxCNC

## 2.1 Introduction

Ce document est centré sur l'utilisation de LinuxCNC, il est plutôt destiné aux lecteurs l'ayant déjà installé et configuré. Quelques informations sur l'installation sont données dans les chapitres suivants. La documentation complète sur l'installation et la configuration se trouve dans le manuel de l'intégrateur.

## 2.2 Comment fonctionne LinuxCNC

LinuxCNC est un peu plus que juste un autre programme de fraiseuse CNC. Il est capable de contrôler des machines-outils, des robots ou d'autres automatismes. Il est capable de contrôler des servomoteurs, des moteurs pas à pas, des relais ainsi que d'autres mécanismes relatifs aux machines-outils.

Il y a quatre principales composantes du logiciel LinuxCNC:
- un contrôleur de mouvement (EMCMOT),
- une contrôleur d'entrées/sorties discrètes (EMCIO),
- un exécuteur des tâches qui les coordonne (EMCTASK),
- et les interfaces utilisateur graphiques.

En outre il existe une couche appelée HAL (couche d'abstraction du matériel) qui permet la configuration de LinuxCNC sans avoir besoin de recompiler.

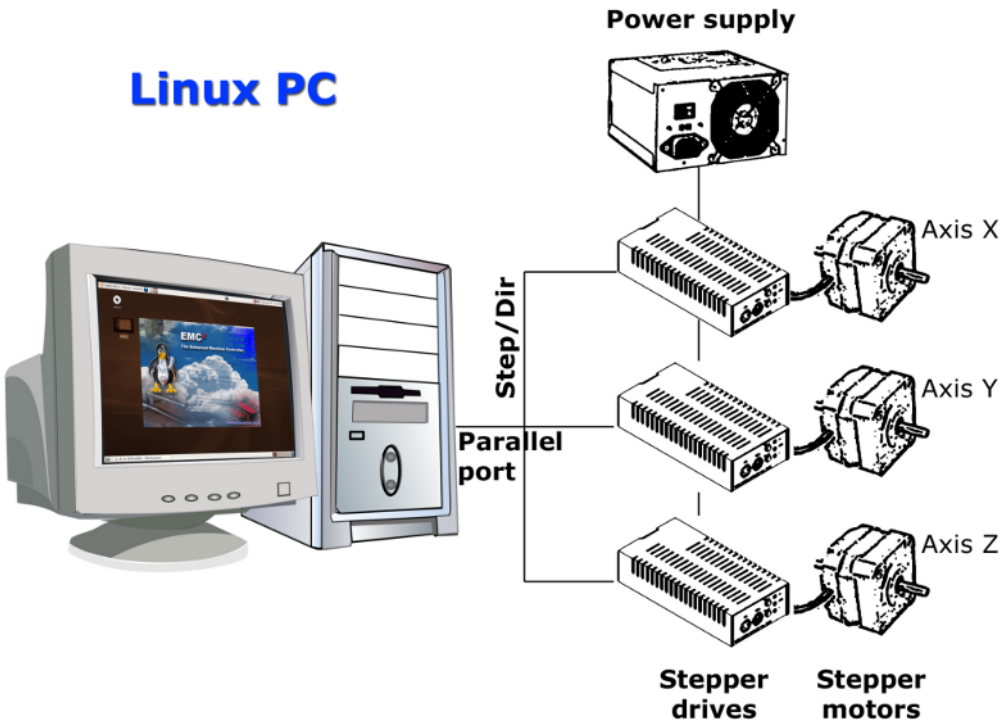



FIGURE 2.1 – Machine simple contrôlée par LinuxCNC

La figure ci-dessus montre un diagramme bloc représentant une machine 3 axes typique comme LinuxCNC les aime. Cette figure montre un système basé sur des moteurs pas à pas. Le PC, tournant sous Linux contrôle les interfaces de puissance des moteurs pas à pas en leur envoyant des signaux au travers du port parallèle. Ces signaux (impulsionnels) font que la puissance adéquate est fournie aux moteurs. LinuxCNC peut également contrôler des servomoteurs via une interface de puissance pour servomoteurs ou utiliser le port parallèle étendu connecté à une carte de contrôle externe. Quand nous examinerons chacun des composants qui forment un système LinuxCNC, nous nous référerons à cette machine typique.

## 2.3 Interfaces utilisateur graphiques

L'interface graphique est la partie de LinuxCNC qui interagit avec l'opérateur de la machine. LinuxCNC est fourni avec plusieurs interfaces utilisateurs graphiques:
– Axis, l'interface utilisateur standard.

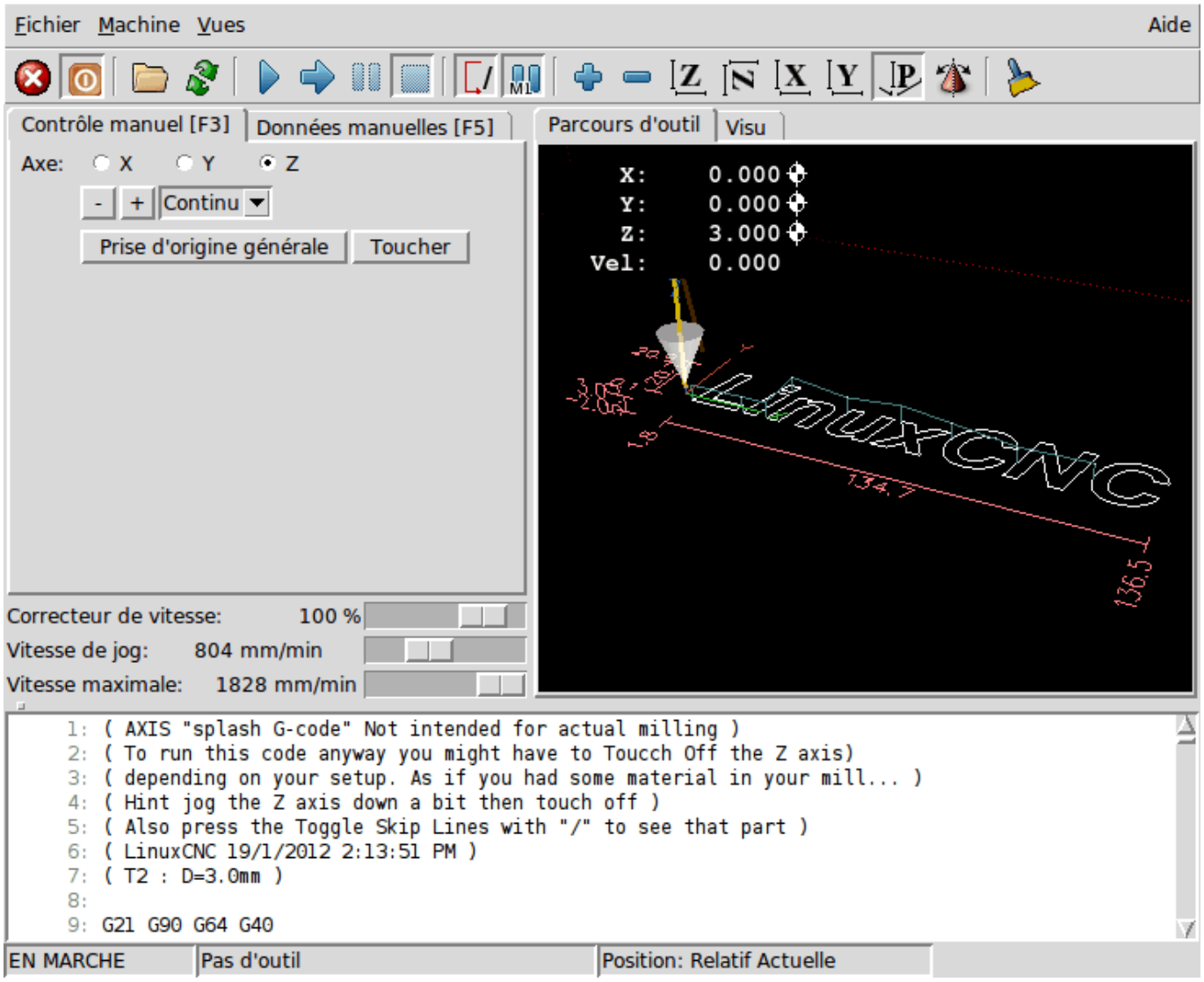



FIGURE 2.2 – L'interface graphique AXIS

– Touchy, une interface graphique pour écran tactile.

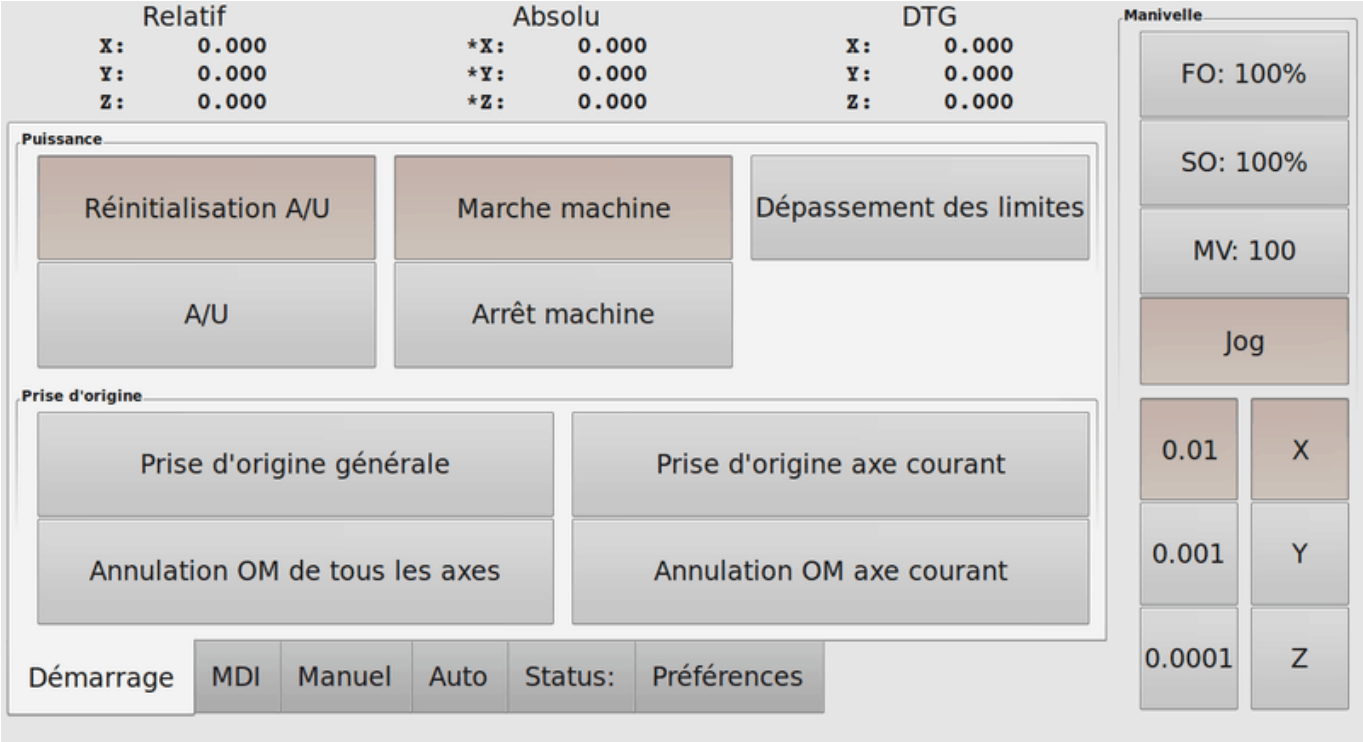



FIGURE 2.3 – L'interface graphique Touchy

– NGCGUI, une interface graphique gérant les sous-programmes. Elle permet très simplement de créer des programme G-code. Elle supporte surtout la concaténation de fichiers de sous-programmes, ce qui permet de construire des programmes G-code complets sans aucune programmation.

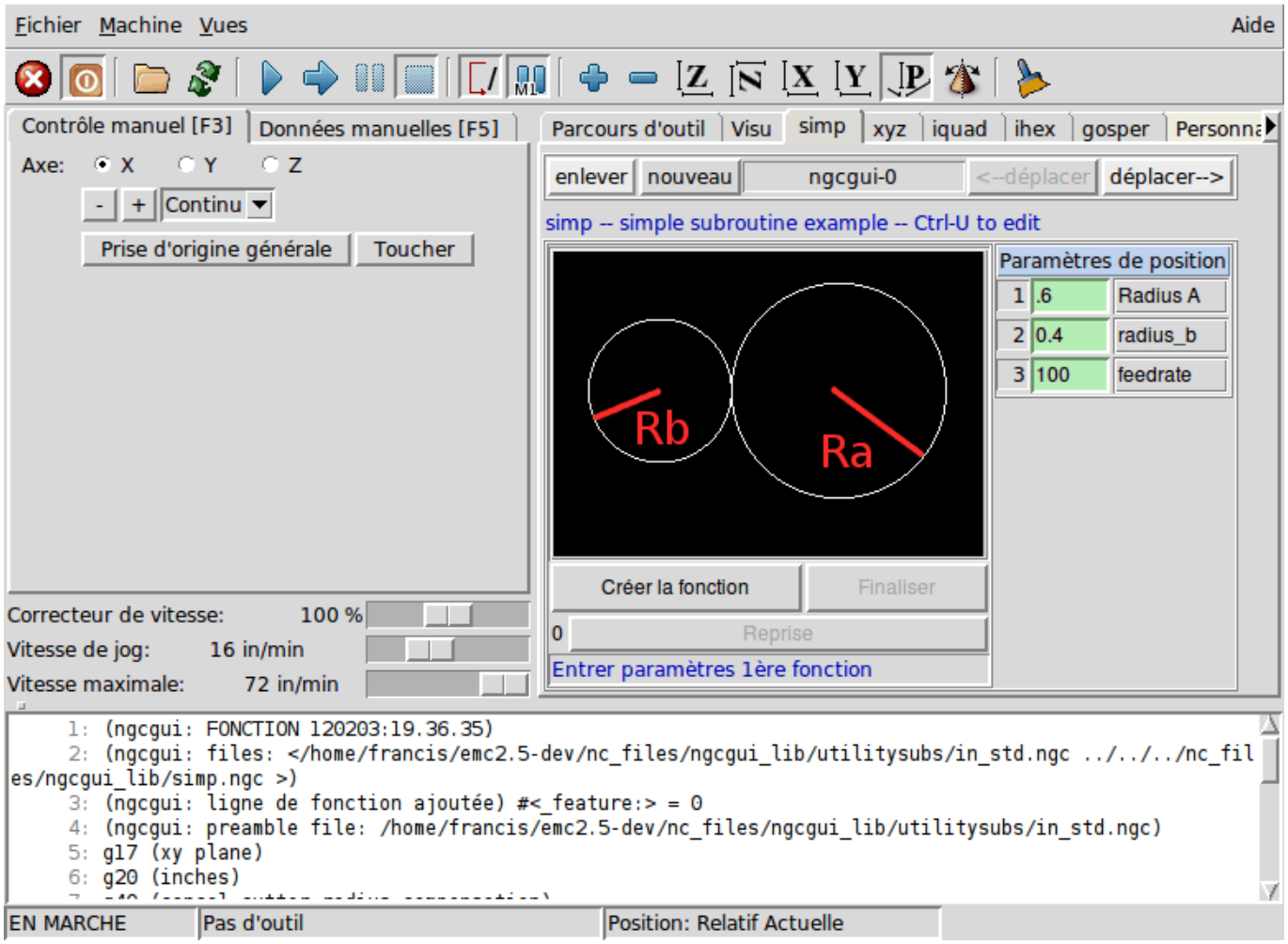



FIGURE 2.4 – L'interface graphique NGCGUI intégrée dans Axis

– Mini, une interface en Tcl/Tk.

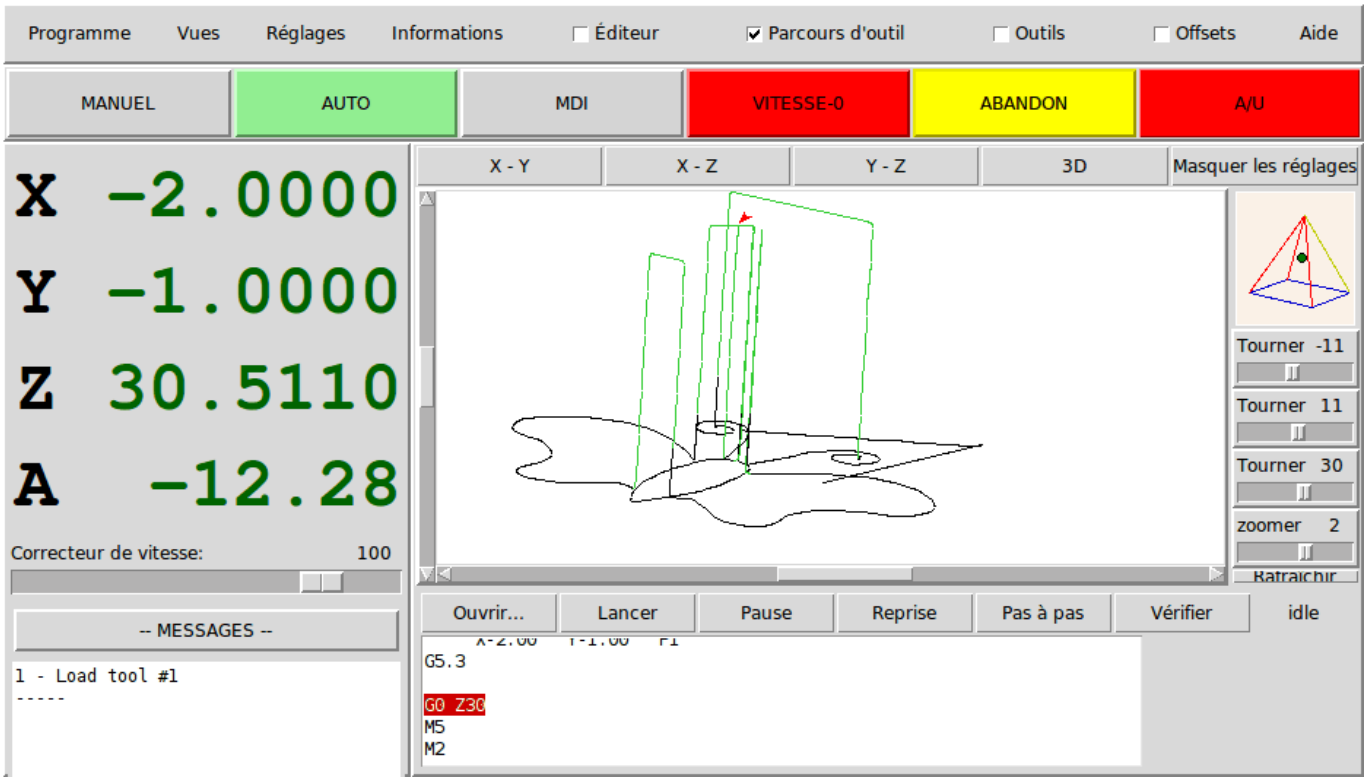



FIGURE 2.5 – L'interface graphique Mini

– TkLinuxCNC, une autre interface basée sur Tcl/Tk. C'est l'interface la plus populaire après Axis

FIGURE 2.6 – L'interface graphique tklinuxcnc

– Keystick, une interface graphique textuelle et minimaliste, appropriée pour les installations légères (sans serveur X par exemple).

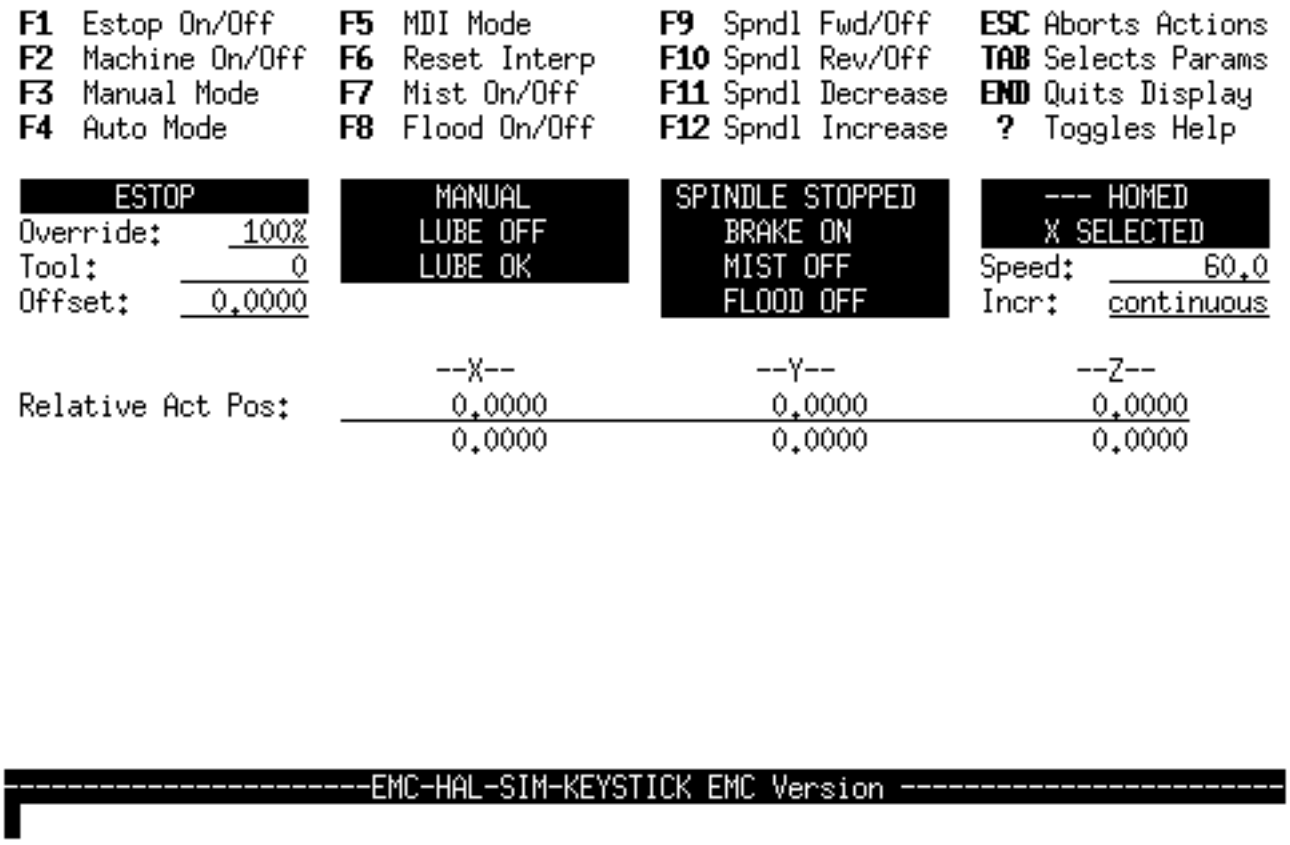



FIGURE 2.7 – L'interface textuelle Keystick

- *Xemc*, un programme X-Windows
- *halui*, une interface utilisateur basée sur HAL, qui permet de contrôler LinuxCNC en utilisant des boutons et des interrupteurs
- *linuxcncrsh*, une interface utilisateur basée sur telnet, qui permet d'envoyer des commandes à partir d'ordinateurs distants de celui de LinuxCNC

## 2.4 Panneaux de contrôle virtuels

- *PyVCP*, un panneau de contrôle virtuel basé sur Python, il peut être intégré dans l'interface graphique Axis ou utilisé en autonome.
- *GladeVCP*, un panneau de contrôle virtuel basé sur Glade, il peut être intégré dans l'interface graphique Axis ou utilisé en autonome.

## 2.5 Langues

LinuxCNC utilise des fichiers traduits pour les interfaces utilisateur. Il fonctionne dans plusieurs langues et démarre dans la langue de la session ouverte par l'utilisateur au démarrage du PC. Si votre langue n'a pas encore été traduite contactez un développeur sur l'IRC ou sur la mailing liste si vous pouvez aider à la traduction.

## 2.6 Penser comme un opérateur sur CNC

Ce manuel ne prétend pas vous apprendre à utiliser un tour ou une fraiseuse. Devenir un opérateur expérimenté prends beaucoup de temps et demande beaucoup de travail. Un auteur a dit un jour, *Nous apprenons par l'expérience, si on la possède toute*. Les

outils cassés, les étaux attaqués et les cicatrices sont les preuves des leçons apprises. Une belle finition, des tolérances serrées et la prudence pendant le travail sont les preuves des leçons retenues. Aucune machine, aucun programme ne peut remplacer l'expérience humaine.

Maintenant que vous commencez à travailler avec le programme LinuxCNC, vous devez vous placer dans la peau d'un opérateur. Vous devez être dans le rôle de quelqu'un qui a la charge d'une machine. C'est une machine qui attendra vos commandes puis qui exécutera les ordres que vous lui donnerez. Dans ces pages, nous donnerons les explications qui vous aideront à devenir un bon opérateur de CNC avec LinuxCNC. Vous aurez besoin de bonnes informations ici, devant vous, c'est là que les pages suivantes prendront tout leur sens.

## 2.7 Modes opératoires

Quand LinuxCNC fonctionne, il existe trois différents modes majeurs pour entrer des commandes. Les modes *Manuel*, *Auto* et *MDI*. Passer d'un mode à un autre marque une grande différence dans le comportement de LinuxCNC. Des choses spécifiques à un mode ne peuvent pas être faites dans un autre. L'opérateur peut faire une prise d'origine sur un axe en mode manuel mais pas en mode auto ou MDI. L'opérateur peut lancer l'exécution complète d'un programme de G-codes en mode auto mais pas en mode manuel ni en MDI.

En mode manuel, chaque commande est entrée séparément. En termes humains, une commande manuelle pourrait être *active l'arrosage* ou *jog l'axe X à 250 millimètres par minute*. C'est en gros, équivalent à basculer un interrupteur ou à tourner la manivelle d'un axe. Ces commandes sont normalement contrôlées en pressant un bouton de l'interface graphique avec la souris ou en maintenant appuyée une touche du clavier. En mode auto, un bouton similaire ou l'appui d'une touche peut être utilisé pour charger ou lancer l'exécution complète d'un programme de G-codes stocké dans un fichier. En mode d'entrée de données manuelles (MDI) l'opérateur peut saisir un bloc de codes est dire à la machine de l'exécuter en pressant la touche *Return* ou *Entrée* du clavier.

Certaines commandes de mouvement sont disponibles et produisent les mêmes effets dans tous les modes. Il s'agit des commandes *Abandon*, *Arrêt d'Urgence* et *Correcteur de vitesse travail* . Ces commandes se dispensent d'explications.

L'interface utilisateur graphique AXIS supprime certaines distinctions entre Auto et les autres modes en rendant automatique la disponibilité des commandes, la plupart du temps. Il rend également floue la distinction entre Manuel et MDI parce que certaines commandes manuelles comme *Toucher*, sont également implémentées en envoyant une commande MDI. Il fait cela en changeant automatiquement le mode qui est nécessaire pour l'action que l'utilisateur a demandé.

# Première partie

# Les interfaces utilisateur

# Chapitre 3

# L'interface graphique AXIS

## 3.1 Introduction

AXIS est une interface utilisateur graphique pour LinuxCNC, il offre un aperçu permanent du parcours de l'outil. Il est écrit en Python, utilise Tk et OpenGL pour afficher son interface graphique.

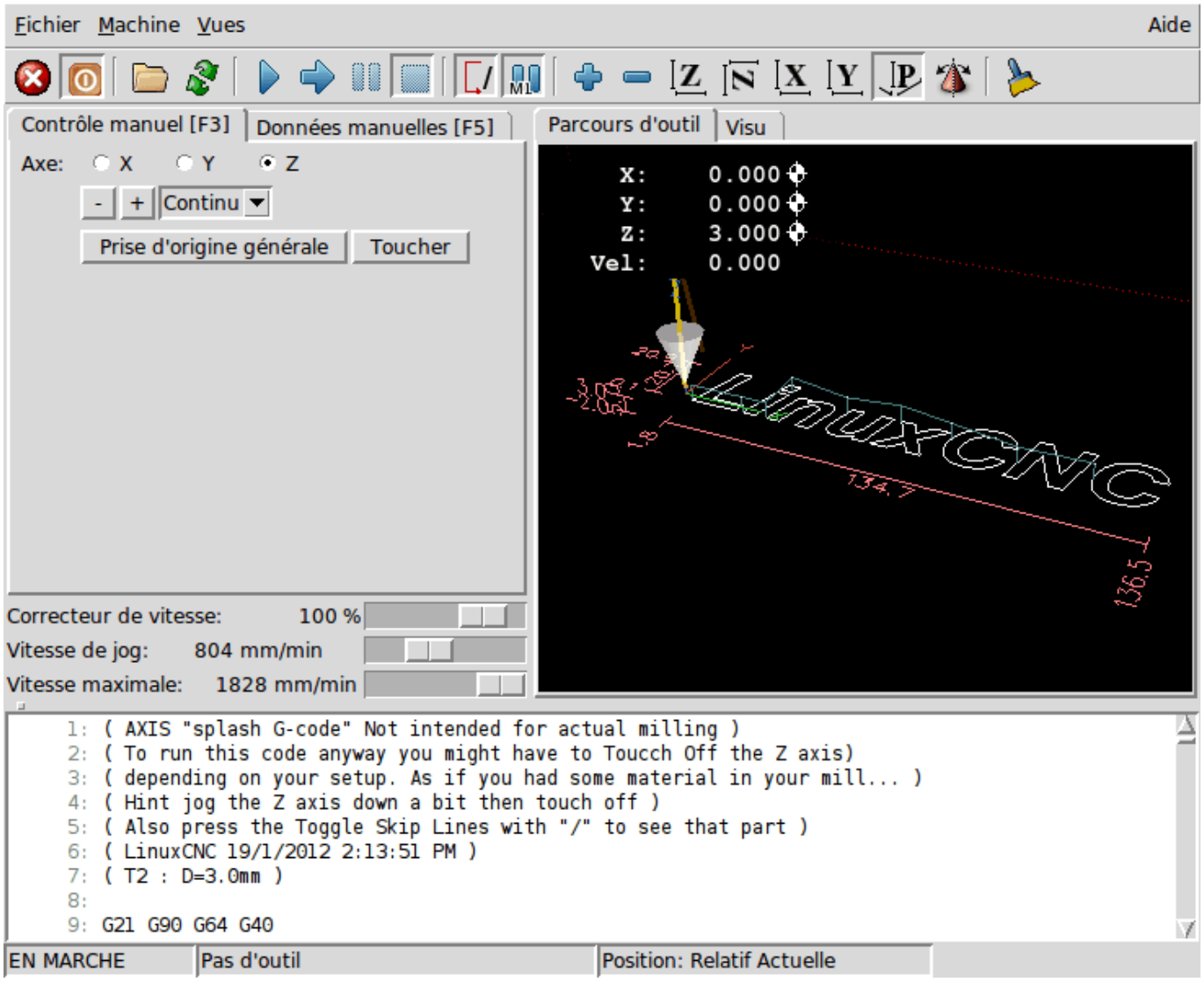



FIGURE 3.1 – Fenêtre d'AXIS

## 3.2 Commencer avec AXIS

Pour choisir AXIS comme interface graphique de LinuxCNC, éditez le fichier .ini et dans la section [DISPLAY] changez la ligne DISPLAY comme ceci:

*DISPLAY = axis*

Puis, lancez LinuxCNC et choisissez le fichier ini. La configuration simplifiée *sim/axis.ini* est déjà configurée pour utiliser AXIS comme interface graphique.

Quand AXIS démarre, une fenêtre telle que celle de la figure ci-dessus s'ouvre.

### 3.2.1 Une session typique avec AXIS

1. Lancer LinuxCNC et sélectionner un fichier de configuration.
2. Relâcher le bouton d'arrêt d'urgence *A/U* et presser celui de *Marche* machine.
3. Faire les prises d'origine machine *POM* pour chacun des axes.

4. Charger un fichier d'usinage.
5. Utiliser l'affichage du parcours d'outil pour vérifier que le programme est correct.
6. Brider le brut à usiner sur la table.
7. Faire les prises d'origine pièce *POP* de chacun des axes avec le jog et en utilisant le bouton *Toucher*.
8. Lancer le programme.
9. Pour usiner le même fichier une nouvelle fois, retourner à l'étape 6. Pour usiner un fichier différent, retourner à l'étape 4. Quand le travail est terminé, quitter AXIS.

## 3.3 Eléments de la fenêtre d'AXIS

La fenêtre d'AXIS contient les éléments suivants:
- Un espace d'affichage qui montre une pré-visualisation du fichier chargé (dans ce cas, *axis.ngc*), ainsi que la position courante du point programmé par la machine. Plus tard, cette zone affichera le parcours de l'outil déplacé par la machine, cette zone est appelée le parcours d'outil (backplot)
- Une barre de menus, une barre d'outils, des curseurs et des onglets permettant d'effectuer différentes actions.
- L'onglet *Contrôle manuel*, qui permet de faire des mouvements d'axe, de mettre la broche en rotation ou de l'arrêter, de mettre l'arrosage en marche ou de l'arrêter.
- L'onglet *Données manuelles* (appelé aussi MDI), où les blocs de programme G-code peuvent être entrés et exécutés à la main, une ligne à la fois.
- Les curseurs *Correcteurs de vitesse*, qui permettent d'augmenter ou de diminuer la vitesse de la fonction concernée.
- Une zone de textes qui affiche le G-code du fichier chargé.
- Une barre d'état qui affiche l'état de la machine. Dans cette capture d'écran, la machine est en marche, aucun outil n'est monté, la position affichée est *relative* à l'origine machine (par opposition à une position *absolue*) et *actuelle* (par opposition à une position *commandée*)

### 3.3.1 Options des menus

Certaines options de menu peuvent s'afficher en grisé, c'est dépendant des options du fichier de configuration ini.

MENU FICHIER
- *Ouvrir...* - C'est une boîte de dialogue standard pour ouvrir un fichier G-code à charger dans AXIS. Si un filtre de programme a été configuré, il peut aussi être ouvert ici.
- *Fichiers récents...* - Affiche la liste des fichiers ouverts récemment.
- *Éditer...* - Ouvre le fichier G-code courant pour édition si un éditeur a été déclaré dans le fichier ini.
- *Recharger...* - Recharge le fichier G-code courant. Si le fichier a été édité, il doit être rechargé pour que les modifications prennent effet. Si un programme a été stoppé, pour le reprendre depuis le début, le recharger. Le bouton *Recharger* a le même effet que l'option de menu.
- *Enregistrer le G-code sous...* - Enregistre le fichier courant sous un nouveau nom.
- *Propriétés...* - Donne la somme des mouvements en vitesse rapide et celle en vitesse travail. Ne tient pas compte des accélérations, ni des décélérations, ni des modes de trajectoire, de sorte qu'il ne donne jamais de temps inférieur au temps réel d'exécution.
- *Éditer la table d'outils...* - Ouvre un dialogue permettant d'éditer les valeurs de la table d'outils.
- *Recharger la table d'outils...* - Après avoir édité la table d'outil, il convient de la recharger pour que les nouvelles valeurs soient prisent en compte.
- *Éditeur de Ladder...* - Si Classic Ladder a été chargé, il est possible de l'éditer ici.
- *Quitter...* - Termine la session courante de LinuxCNC.

MENU MACHINE
- *Arrêt d'Urgence F1...* - (bascule) Active/désactive l'arrêt d'urgence.
- *Marche/Arrêt F2...* - (bascule) Active/désactive la puissance machine.
- *Démarrer le programme...* - Lance l'exécution du programme G-code.
- *Démarrer à la ligne sélectionnée...* - Prudence avec cette commande, respecter la démarche suivante: Premièrement, sélectionner à la souris, la ligne à laquelle démarrer. Déplacer ensuite manuellement, l'outil à la position de la ligne précédente puis, cette commande exécutera le reste du code.

**AVERTISSEMENT**
Ne pas utiliser la commande *Démarrer à la ligne sélectionnée...* si le programme G-code contient des sous-programmes.

- *Pas à pas...* - Avance d'un seul pas de programme.
- *Pause...* - Effectue une pause dans le programme.
- *Reprise...* - Reprends la marche après une pause.
- *Stopper...* - Stoppe le programme en marche.
- *Arrêt sur M1...* - Si M1 est rencontré et que cette option est cochée, l'exécution du programme s'interrompra à la ligne où il a été rencontré. Presser *Reprise* pour continuer.
- *Sauter les lignes avec "/"...* - Si une ligne commençant par / est rencontrée et que cette option est cochée, cette ligne sera sautée.
- *Vider l'historique du MDI...* - Efface l'historique des données manuelles.
- *Copier depuis l'historique du MDI...* - Copier l'historique des données manuelles dans le presse-papier.
- *Coller dans l'historique du MDI...* - Coller le contenu du presse-papier dans la fenêtre d'historique des données manuelles.
- *Calibration* - Lance l'assistant de réglage de PID Servo. La calibration lit le fichier HAL et pour chaque *pas* il utilise une variable de la section [AXIS_n] du fichier ini et crée une entrée pouvant être éditée et testée.
- *Afficher configuration de HAL...* - Ouvre une fenêtre sur la configuration de HAL depuis laquelle il est possible de visualiser tous les *Components*, *Pins*, *Parameters*, *Signals*, *Functions* et *Threads* de HAL.
- *HAL Mètre...* - Ouvre une fenêtre dans laquelle il est possible de visualiser un seul *Signal, HAL Pin*, ou *Parameter* de HAL.
- *HAL Scope...* - Ouvre un oscilloscope virtuel qui permet de tracer dans le temps, les valeurs de HAL.
- *Afficher l'état de LinuxCNC...* - Ouvre une fenêtre montrant l'état de LinuxCNC.
- *Choisir le niveau de Debug...* - Ouvre une fenêtre dans laquelle les niveaux de débogage sont visibles et certains réglables.
- *Prise d'origine...* - Effectue la prise d'origine machine d'un ou de tous les axes.
- *Annulation OM...* - Annule les origines d'un ou de tous les axes.
- *Annulation décalages d'origine...* - Annule les décalages d'origine du système de coordonnées choisi.

- *L'outil touchera la pièce...* - Lorsqu'un *Toucher* est effectué, la valeur entrée est relative au système de coordonnées pièce actuel (G5x), tel que modifié par le décalage d'axe (G92). Quand la séquence de *Toucher* est complète, la coordonnée relative pour l'axe choisi prendra la valeur entrée. Voir aussi G10 L10 dans le chapitre du G-code.

  **L'outil touchera le porte-pièce...**
  Lorsqu'un *Toucher* est effectué, la valeur entrée est relative au 9ème système de coordonnées (G59.3), le décalage d'axe (G92) est ignoré. Mode destiné aux machines possédant un porte-pièce référencé à un endroit, sur lequel s'effectue le *Toucher*. Le 9ème système de coordonnées doit être ajusté pour que la pointe d'un outil de longueur nulle (le nez de broche), soit à l'origine du porte-pièce quand les coordonnées relatives sont à 0. Voir aussi G10 L11 dans le chapitre du G-code.

**Tout est dans le point de vue...**

Les icônes de choix du type d'affichage et du menu *Vues* d'AXIS se référent à des *Vue de dessus*, *Vue de face* et *Vue de côté*. Ces termes sont corrects si la machine CNC a un axe Z vertical, avec une valeur de Z positive en haut. C'est vrai pour les fraiseuses verticales, qui sont probablement les plus populaires, c'est également vrai pour toutes les machines d'électro-érosion et aussi les tours verticaux, sur lesquels la pièce tourne sous l'outil.

Les termes *Vue de dessus*, *Vue de face* et *Vue de côté* sont cependant source de confusion sur d'autres machines CNC, comme un tour standard, sur lequel l'axe Z est horizontal, ou sur une fraiseuse horizontale, qui a également l'axe Z horizontal, ou même un tour vertical inversé, sur lequel la pièce tourne au dessus de l'outil et qui a son axe Z positif vers le bas!

Il faut juste se rappeler que l'axe Z est toujours parallèle a la broche et plus positif en s'éloignant de celle-ci. Etre familiarisé avec la cinématique de ses machines, permet d'interpréterez l'affichage comme il se doit.

- *Vue de dessus...* - La vue de dessus (ou vue de Z) affiche l'aspect du G-code vu depuis le côté positif de l'axe Z et en regardant vers son côté négatif. Cette vue convient bien pour visualiser les axes X et Y.
- *Vue de dessus basculée...* - La vue de dessus basculée (ou vue de Z basculé) affiche également l'aspect du G-code vu depuis le côté positif de l'axe Z et en regardant vers son côté négatif. Mais cette fois, les axes X et Y sont représentés pivotés de 90 degrés pour mieux occuper l'espace d'affichage. Cette vue convient bien également, pour visualiser les axes X et Y.

- *Vue de côté...* - La vue de côté (ou vue de X) affiche l'aspect du G-code vu depuis le côté positif de l'axe X et en regardant vers son côté négatif. Cette vue convient pour visualiser les axes Y et Z.
- *Vue de face...* - La vue de face (ou vue de Y) affiche l'aspect du G-code vu depuis le côté positif de l'axe Y et en regardant vers son côté négatif. Cette vue convient bien pour visualiser les axes X et Z.
- *Vue en perspective...* - La vue en perspective (ou vue P) affiche l'aspect du G-code en regardant vers la pièce depuis un point de vue orientable, par défaut vers X+, Y-, Z+. Cette position est orientable en la sélectionnant à la souris. L'affichage est un compromis, il tente d'afficher en 3D, entre trois et neuf axes, sur un écran en deux dimensions. Il y aura donc souvent certaines caractéristiques difficiles à voir, ce qui requerra un changement de point de vue. Cette vue convient bien pour voir les trois axes à la fois.
- *Affichage en pouces...* - Ajuste l'échelle d'affichage d'AXIS pour les pouces.
- *Affichage en mm...* - Ajuste l'échelle d'affichage d'AXIS pour les millimètres.
- *Afficher le programme...* - L'affichage à l'écran de l'aspect du G-code peut être entièrement désactivé si l'opérateur le souhaite.
- *Parcours d'outil en vitesse rapide...* - L'affichage du parcours d'outil du programme G-code courant représente toujours les mouvements en vitesse travail (G1,G2,G3) en blanc. Mais l'affichage des mouvements en vitesse rapide (G0) en cyan peut être désactivé si si l'opérateur le souhaite.
- *Simulation de transparence...* - Cette option rends plus lisible le tracé des parcours affichés par les programmes complexes, mais il peut rendre l'affichage plus lent.
- *Parcours d'outil en temps réel...* - La surbrillance des chemins d'outils en vitesse travail (G1,G2,G3) quand l'outil se déplace peut être désactivée si l'opérateur le souhaite.
- *Afficher l'outil...* - Le symbole d'un outil, représenté par un cône ou un cylindre peut être désactivé si l'opérateur le souhaite.
- *Afficher les étendues...* - L'affichage des étendues du programme G-code chargé (déplacements maximum de chacun des axes), peut être désactivé si l'opérateur le souhaite.
- *Afficher les offsets...* - L'emplacement de l'origine du système de coordonnées pièce (G54 à G59.3) peut être représenté par un jeu de trois lignes orthogonales, une rouge, une bleue et une verte. L'affichage de cette origine pièce (ou zéro pièce), peut être désactivé si l'opérateur le souhaite.
- *Afficher les limites machine...* - Les limites maximales de déplacement machine pour chacun des axes, qui sont fixées dans le fichier ini, s'affichent comme une boîte rectangulaire en lignes pointillées rouges. Il est facile, au chargement d'un nouveau programme G-code, de voir si la pièce est contenue dans le volume représenté. Ou de vérifier de combien l'étau doit être décalé, pour que le G-code puisse être usiné sans dépasser les limites de déplacements de la machine. Cette option peut être désactivée si l'opérateur le souhaite.
- *Afficher la vitesse d'avance...* - L'affichage de la vitesse peut être utile pour voir la précision avec laquelle la machine suit la vitesse commandée. Cette option peut être désactivée si l'opérateur le souhaite.
- *Afficher la distance restante...* - La distance restante est une valeur très utile à suivre, au lancement d'un programme de G-code inconnu pour la première fois. En combinaison avec les curseurs des correcteurs de vitesse, des dégâts sur l'outil ou la machine peuvent être évités. Quand le programme G-code sera débogué et qu'il fonctionnera en douceur, l'affichage de la distance restante pourra être désactivée si l'opérateur le souhaite.
- *Coordonnées en police large...* - Les coordonnées des axes et la vitesse d'avance, s'afficheront en police large dans la vue du parcours d'outil.
- *Rafraîchir le parcours d'outil...* - Au fur et à mesure des déplacements de l'outil, les parcours s'affichent sur l'écran d'Axis en surbrillance. Avant de répéter le programme, ou pour avoir un affichage clair sur une zone intéressante, la surbrillance des parcours précédents peut être rafraîchie.
- *Afficher la position commandée...* - C'est la position que LinuxCNC cherche à atteindre. Quand le mouvement est stoppé, c'est la position que LinuxCNC cherchera à maintenir.
- *Afficher la position actuelle...* - La position actuelle est la position mesurée grâce aux informations issues des codeurs ou simulées par le générateur de pas. Elle peut différer légèrement de la position commandée pour diverses raisons, comme les réglages des boucles PID, les contraintes physiques ou les efforts de coupe.
- *Afficher la position machine...* - C'est la position par rapport à l'origine machine, telle qu'établie par la prise d'origine machine *(POM)*.
- *Afficher la position relative...* - C'est la position par rapport à l'origine pièce, telle qu'établie par la prise d'origine pièce *(POP)*. On peut aussi représenter cette position comme étant l'origine machine à laquelle on a appliqué les codes de décalages des systèmes de coordonnées G5x, G92 et G43.

MENU AIDE

- *A propos d'Axis...* - Donne la version et quelques informations relatives au copyright.
- *Aide rapide...* - Affiche la liste des raccourcis clavier.

### 3.3.2 Boutons de la barre d'outils

Signification des boutons de la fenêtre d'AXIS, de gauche à droite:

- *Arrêt d'urgence* (A/U) (en Anglais, E-Stop)
- Marche/Arrêt puissance machine
- Ouvrir un fichier
- Recharger le fichier courant
- Départ cycle
- Cycle en pas à pas
- Pause/Reprise
- Stopper l'exécution du programme
- Sauter ou non les lignes commençant par **/**
- Avec ou sans pause optionnelle
- Zoom plus
- Zoom moins
- Vue prédéfinie **Z** (vue de dessus)
- Vue prédéfinie **Z basculée**
- Vue prédéfinie **X** (vue de côté)
- Vue prédéfinie **Y** (vue de face)
- Vue prédéfinie **P** (vue en perspective)
- Orienter la vue avec le bouton gauche de la souris
- Rafraîchir le parcours d'outil

### 3.3.3 Zones d'affichage graphique du programme

**Affichage des coordonnées** L'affichage des coordonnées est situé en haut à gauche de l'écran graphique. Il montre les positions de la machine. A gauche du nom de l'axe, un symbole d'origine est visible si la prise d'origine de l'axe a été faite.

Symbole de prise d'origine faite.

A droite du nom de l'axe, un symbole de limite est visible si l'axe est sur un de ses capteurs de limite.

Symbole de limite d'axe.

Pour interpréter correctement ces valeurs, référez vous à l'indicateur *Position* de la barre d'état. Si la position est *Absolue*, alors les valeurs affichées sont exprimées en coordonnées machine. Si la position est *Relative*, alors les valeurs affichées sont exprimées

en coordonnées relatives à la pièce. Quand les coordonnées affichées sont relatives, une marque d'origine de couleur cyan est visible pour représenter l'origine machine.

Symbole d'origine machine.

Si la position est *Commandée*, alors il s'agit de la position à atteindre. Par exemple, les coordonnées passées dans une commande **G0**. Si la position est *Actuelle*, alors il s'agit de la position à laquelle la machine vient de se déplacer. Ces valeurs peuvent varier pour certaines raisons: erreur de suivi, bande morte, résolution d'encodeur, ou taille de pas. Par exemple, si vous demandez un mouvement à X 0.08 à votre fraiseuse, mais un pas du moteur fait 0.03, alors la position *Commandée* sera de 0.08, mais la position *Actuelle* sera de 0.06 (2 pas) ou 0.09 (3 pas).

**Vue du parcours d'outil**

Quand un fichier est chargé, une vue du parcours d'outil qu'il produira est visible dans la zone graphique. Les mouvements en vitesse rapide (tels ceux produits par une commande **G0**) sont affichés en lignes pointillées vertes. Les déplacements en vitesse travail (tels ceux produits par une commande **G1**) sont affichés en lignes continues blanches. Les arrêts temporisés (tels ceux produits par la commande **G4**) sont représentés par une petite marque **X**.

Un mouvement G0 (Vitesse rapide) avant un déplacement en vitesse travail ne sera pas affiché sur l'écran des parcours d'outil. Un mouvement en vitesse rapide, après un appel d'outil T<n>, n'apparaîtra sur l'écran des parcours d'outil qu'après le mouvement en vitesse travail suivant. Pour contourner une de ces caractéristiques, programmer un G1 sans déplacement, juste avant le G0.

**Étendues du programme**

Les *étendues* du programme sont affichées pour chacun des axes. Aux extrémités, les coordonnées minimales et maximales sont indiquées. Au centres, la différence, entre ces deux coordonnées, est indiquée.

Quand une coordonnée dépasse la limite logicielle fixée dans le fichier .ini, la coordonnée correspondante s'affiche en rouge, entourée d'un rectangle. Dans la figure ci-dessous, la limite maximale est dépassée sur l'axe X, comme l'indique le rectangle entourant la valeur de la coordonnée. Le déplacement X minimal du programme est de -1.95, la course maximale est de 1.88 en X et le programme nécessite un déplacement en X de 3.83 pouces. Le déplacement total demandé par le programme est donc possible. Pour cela, se déplacer en jog vers la gauche puis *Toucher* à nouveau pour corriger l'origine pièce.

FIGURE 3.2 – Limites logicielles

**Le cône d'outil** Si aucun outil n'est chargé, l'emplacement de la pointe de l'outil est indiqué par le *cône d'outil*. Le cône d'outil ne donne aucune indication sur la forme, la longueur, ou le rayon de l'outil.

Quand un outil est chargé, par exemple dans le MDI, avec la commande **T1 M6**, le cône d'outil passe de conique à cylindrique, il indique alors la proportion du diamètre de l'outil lu dans le fichier de la table d'outils.

**Parcours d'outil** Quand la machine se déplace, elle laisse une trace appelée le parcours d'outil. La couleur des lignes indique le type de mouvement: jaune pour les mouvementq jog, vert clair pour les mouvements en vitesse rapide, rouge pour les mouvements en vitesse d'avance programmée et magenta pour les mouvements circulaires en vitesse d'avance programmée.

**Interaction avec l'affichage** Par un clic gauche sur une portion du parcours d'outil, la ligne sous la souris passe en surbrillance à la fois dans le parcours d'outil et dans le texte. Un clic droit dans une zone vide enlève la surbrillance

En déplaçant la souris avec son bouton gauche appuyé, la vue est glissée sur l'écran.

En déplaçant la souris avec le bouton *Maj* enfoncé, ou en glissant avec la molette de la souris appuyée, la vue est tournée. Si une ligne du tracé est en surbrillance, elle devient le centre de rotation de la vue. Autrement, le centre de rotation est le milieu du fichier dans son ensemble.

En tournant la molette de la souris, ou en glissant la souris avec son bouton droit enfoncé, ou encore en glissant la souris avec son bouton gauche enfoncé et la touche *Ctrl* appuyée, le tracé sera zoomé en plus ou en moins.

En cliquant sur une des icônes de vue pré-définie de la barre d'outils, ou en pressant la touche **V**, cette vue est sélectionnée.

### 3.3.4 Zone texte d'affichage du programme

Un clic gauche sur une ligne du programme passe la ligne en surbrillance à la fois dans la zone texte et dans le parcours d'outil.

Quand le programme est lancé, la ligne en cours d'exécution est en surbrillance rouge. Si aucune ligne n'est sélectionnée par l'utilisateur, le texte défile automatiquement pour toujours laisser la ligne courante visible.

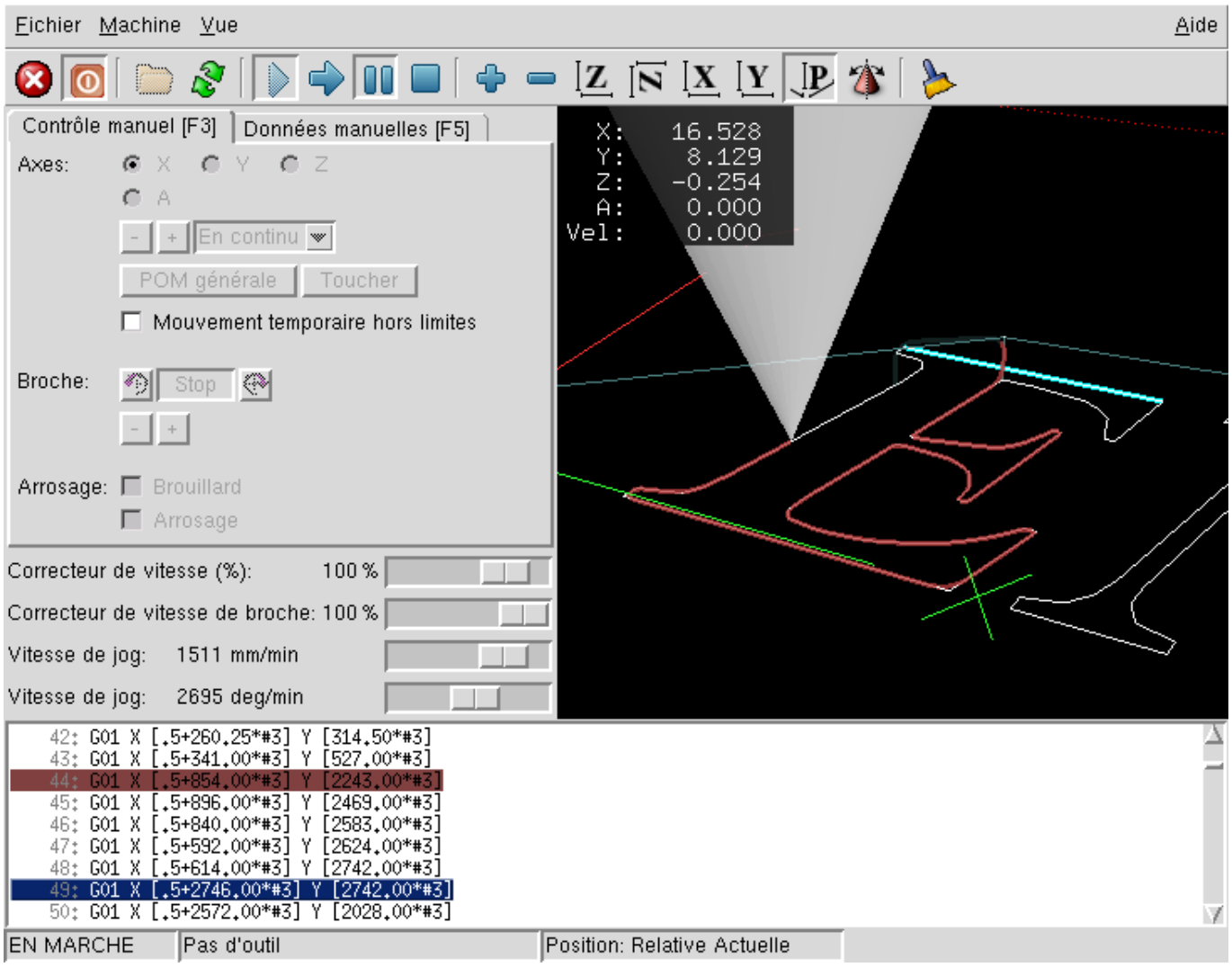



FIGURE 3.3 – Ligne courante et ligne en surbrillance

### 3.3.5 Contrôle manuel

Quand la machine est en marche mais qu'aucun programme n'est exécuté, les éléments graphiques de l'onglet *Contrôle manuel* peuvent être utilisés pour actionner la machine ou mettre en marche et arrêter ses différents organes.

Quand la machine n'est pas en marche, ou quand un programme est en cours d'exécution, le contrôle manuel n'est pas disponible.

Certains des éléments décrits plus bas ne sont pas disponibles sur toutes les machines. Quand AXIS détecte qu'une pin particulière n'est pas connectée dans le fichier HAL, l'élément correspondant de l'onglet *Contrôle manuel* est supprimé. Par exemple, si la pin HAL *motion.spindle-brake* n'est pas connectée, alors le bouton *Frein de broche* n'apparaîtra pas sur l'écran. Si la variable d'environnement AXIS_NO_AUTOCONFIGURE est mise à 1, ce comportement est désactivé et tous les boutons sont visibles.

**Le groupe de cases et boutons Axes** Les cases à cocher du groupe *Axes* permettent de choisir l'axe de la machine à actionner manuellement. Cette action s'appelle le *jog*. Premièrement sélectionner l'axe à actionner en cochant sa case. Puis cliquer sur le bouton **+** ou **-** selon le sens de déplacement souhaité. Les quatre premiers axes peuvent aussi être déplacés avec les touches fléchées pour X et Y, avec les touches Page précédente et Page suivante pour (Z) et les touches [ et ] pour A.

Si *En continu* est sélectionné, le mouvement continuera tant que la touche ou le bouton resteront appuyés. Si une autre valeur est sélectionnée, la machine se déplacera juste de la distance affichée à chaque fois que la touche ou le bouton seront appuyés. Par défaut, les valeurs disponibles sont:

```
0.1000 0.0100 0.0010 0.0001
```

Voir le Manuel de l'intégrateur pour plus d'informations sur la configuration des incréments de jog.

**Prise d'origine machine** Si votre machine dispose de contacts d'origine machine et a une séquence de prise d'origine définie dans le fichier ini, le bouton *POM générale* lancera cette séquence pour tous les axes, les touches *Ctrl-HOME* auront le même effet.

Si votre machine dispose de contacts d'origine mais n'a pas de séquence de prise d'origine définie dans le fichier ini, le bouton *POM générale* effectuera uniquement la prise d'origine de l'axe sélectionné. Cette procédure doit alors être réalisée, séparément pour chacun des axes.

Si votre machine ne dispose d'aucun contact d'origine défini dans la configuration, le bouton *POM générale* définira la position actuelle de l'axe comme étant la position d'origine machine et l'axe sera marqué comme ayant sa prise d'origine machine faite.

**Toucher**

Si le bouton *Toucher* ou la touche *FIN* sont appuyés, le décalage d'origine pièce de l'axe Z, sur la figure ci-dessous: P1 G54, prendra la valeur spécifiée dans le champ de la boite de dialogue. Les expressions peuvent être entrées en suivant les règles de programmation rs274ngc, sauf les variables qui ne peuvent pas être utilisées. La valeur résultante sera affichée sous le champ. Exemple, pour faire la prise d'origine pièce, on affleure l'outil sur une cale de 5mm d'épaisseur posée sur le bloc, on presse le bouton *Toucher* et on saisi 5 dans le champ de la boîte de dialogue. La pointe de l'outil sera alors référencée à 0 sur la surface du bloc.

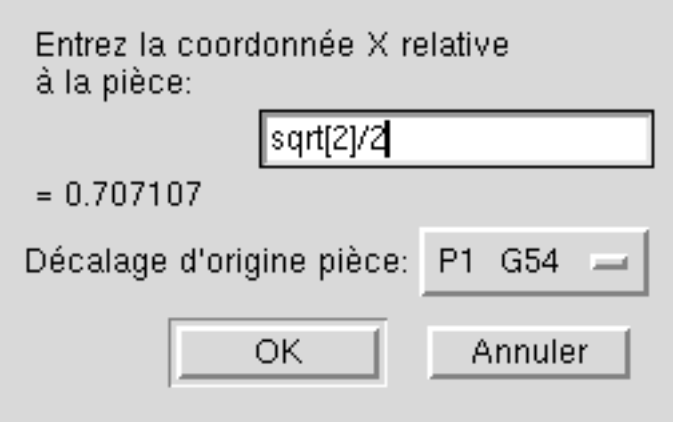



FIGURE 3.4 – Fenêtre du Toucher

Voir aussi les options du menu Machine: *Toucher la pièce* et *Toucher le porte-pièce*.

**Dépassement de limite** En appuyant sur *Dépassement de limite*, la machine sera temporairement autorisée à se déplacer au delà d'un contact de limite physique. Cette case à cocher n'est disponible que lorsque un fin de course est pressé. Elle est désactivée après chaque mouvement de jogging. Si l'axe est configuré avec des contacts positifs et négatifs séparés, LinuxCNC permettra

le jogging uniquement dans le sens du dégagement. *Dépassement de limite* ne permettra pas un jogging au delà d'une limite logicielle. La seule façon de désactiver une limite logicielle sur un axe est d'annuler sa prise d'origine.

**Le groupe Broche**

Les boutons de la première rangée permettent de sélectionner la direction de rotation de la broche: Sens anti-horaire, Arrêt, Sens horaire. Les boutons de la rangée suivante augmentent ou diminuent la fréquence de rotation. La case à cocher de la troisième rangée permet d'engager ou de relâcher le frein de broche. Selon la configuration de votre machine, ces éléments n'apparaîtront peut être pas tous.

**Le groupe Arrosage**

Ces deux boutons permettent d'activer les *gouttelettes* et l'*Arrosage fluide* ou de les désactiver. Selon la configuration de votre machine, ces boutons n'apparaîtront peut être pas tous.

### 3.3.6 Données manuelles (MDI)

L'onglet d'entrée de données manuelles (encore appelé MDI), permet d'entrer et d'exécuter manuellement et une par une, des lignes de programme en G-code. Quand la machine n'est pas en marche, ou quand un programme est en cours d'exécution, cet onglet n'est pas opérationnel.

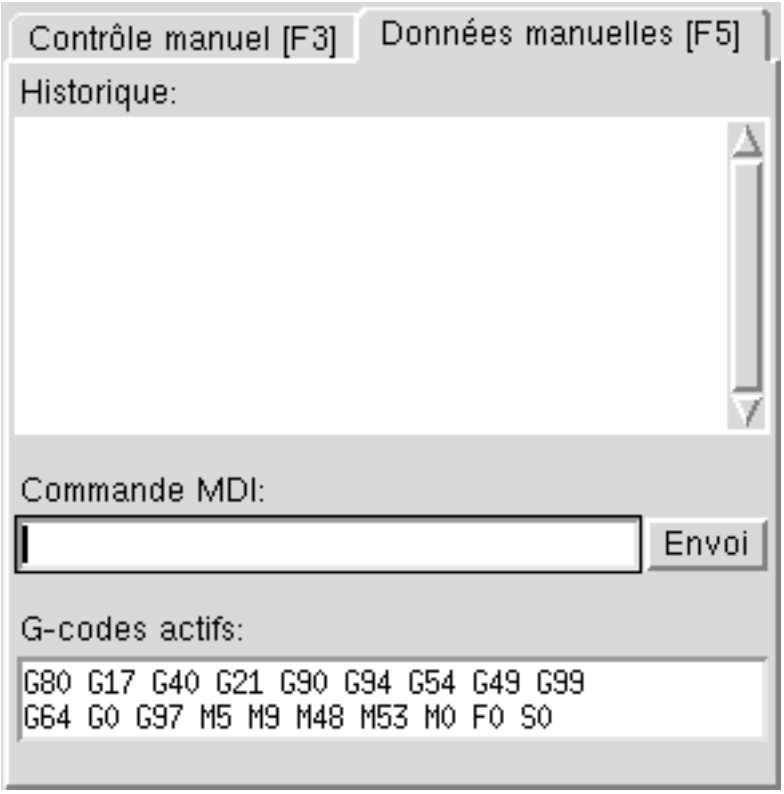



FIGURE 3.5 – L'onglet *Données manuelles*

- *Historique* - Affiche les commandes précédemment tapées et au cours des session précédentes.
- *Commande MDI* - Ce champ permet la saisie d'une ligne de commande à exécuter. La commande sera exécutée par l'appui de la touche <Entrée> ou un appui sur le bouton *Envoi*.
- *G-Codes actifs* - Cette fenêtre affiche les *codes modaux* actuellement actifs dans l'interpréteur. Par exemple, **G54** indique que le système de décalage d'origine actuel est **G54** qui s'appliquera à toutes les coordonnées qui seront entrées.

### 3.3.7 Correcteurs de vitesse

En déplaçant le curseur, la vitesse de déplacement programmée peut être modifiée. Par exemple, si un programme requiert une vitesse à **F600** et que le curseur est placé sur 120%, alors la vitesse résultante sera de **F720**.

### 3.3.8 Correcteur de vitesse de broche

En déplaçant ce curseur, la vitesse programmée de la broche peut être modifiée. Par exemple, si un programme requiert une vitesse à F8000 et que le curseur est placé sur 80%, alors la fréquence de rotation résultante sera de **F6400**. Cet élément n'apparaît que si la *HAL pin motion.spindle-speed-out* est connectée dans le fichier .hal.

### 3.3.9 Vitesse de Jog

En déplaçant ce curseur, la vitesse de jog peut être modifiée. Par exemple, si ce curseur est placé sur 100 mm/mn, alors un jog de 1 mm durera .6 secondes, ou 1/100 de minute. Du côté gauche du curseur (jog lent) l'espacement des valeurs est petit alors que du côté droit (jog rapide) l'espacement des valeurs est plus grand, cela permet une large étendue de vitesses de jog avec un contrôle plus fin du curseur dans les zones les plus importantes.

Sur les machines avec axes rotatifs, un second curseur de vitesse est présent. Il permet d'ajuster la vitesse de rotation des axes rotatifs (A, B et C).

### 3.3.10 Vitesse Maxi

En déplaçant ce curseur, la vitesse maximale peut être réglée. Ceci couvre la vitesse maximale de tous les mouvements programmés, sauf les mouvements avec broche synchronisée.

## 3.4 Raccourcis clavier

La plupart des actions d'AXIS sont accessibles depuis le clavier. La liste complète des raccourcis clavier est disponible dans l'aide rapide d'AXIS qui s'affiche en cliquant sur Aide > Aide rapide . Beaucoup de ces raccourcis sont inaccessible en mode Entrées manuelles.

Touches des correcteurs de vitesse.
- Les touches des correcteurs de vitesse se comportent différemment en mode manuel. Les touches *12345678* sélectionneront l'axe correspondant, si il est programmé.
- Si vous avez 3 axes, alors * sélectionnera l'axe 0, 1 sélectionnera l'axe 1, et 2 sélectionnera l'axe 2.
- Pendant l'exécution d'un programme, les touches *1234567890* fixeront la correction de vitesse travail entre 10% et 100%.

Les raccourcis clavier les plus fréquents sont visibles dans la table ci-dessous.

TABLE 3.1: Raccourcis clavier usuels

| **Touches** | **Actions produites** | **Mode** |
|---|---|---|
| F1 | Bascule l'arrêt d'urgence | Tous |
| F2 | Bascule le marche/arrêt machine | Tous |
| `, 1 .. 9, 0 | Correcteurs de vitesse de 10% à 100% | Varie |
| X, * | Active le premier axe | Manuel |
| Y, 1 | Active le deuxième axe | Manuel |
| Z, 2 | Active le troisième axe | Manuel |
| A, 3 | Active le quatrième axe | Manuel |
| I | Sélection d'incrément du jog | Manuel |
| C | jog en mode continu | Manuel |
| Ctrl+origine | Lance une séquence de POM | Manuel |

TABLE 3.1: (continued)

| Touches | Actions produites | Mode |
|---|---|---|
| Fin | Toucher: valide l'offset G54 de l'axe actif | Manuel |
| Gauche, Droite | Jog du premier axe | Manuel |
| Up, Down | Jog du deuxième axe | Manuel |
| Pg Up, Pg Dn | Jog du troisième axe | Manuel |
| [, ] | Jog du quatrième axe | Manuel |
| O | Ouvrir un fichier | Manuel |
| Ctrl+R | Recharger le fichier courant | Manuel |
| R | Exécuter le programme | Manuel |
| P | Pause dans l'exécution du programme | Auto |
| S | Reprise de l'exécution du programme | Auto |
| ESC | Stopper l'exécution | Auto |
| Ctrl+K | Rafraichi le tracé d'outil | Auto/Manuel |
| V | Défilement cyclique des vues prédéfinies | Auto/Manuel |
| Maj-gauche,droite | Axe X vitesse rapide | Manuel |
| Maj-haut, bas | Axe Y vitesse rapide | Manuel |
| Maj-PgUp, PgDn | Axe Z vitesse rapide | Manuel |

## 3.5 Affichage de l'état de LinuxCNC (LinuxCNCtop)

AXIS inclut un programme appelé *linuxcnctop* qui affiche en détail l'état de LinuxCNC. Ce programme est accessible dans le menu Machine > Fenêtre d'état de LinuxCNC.

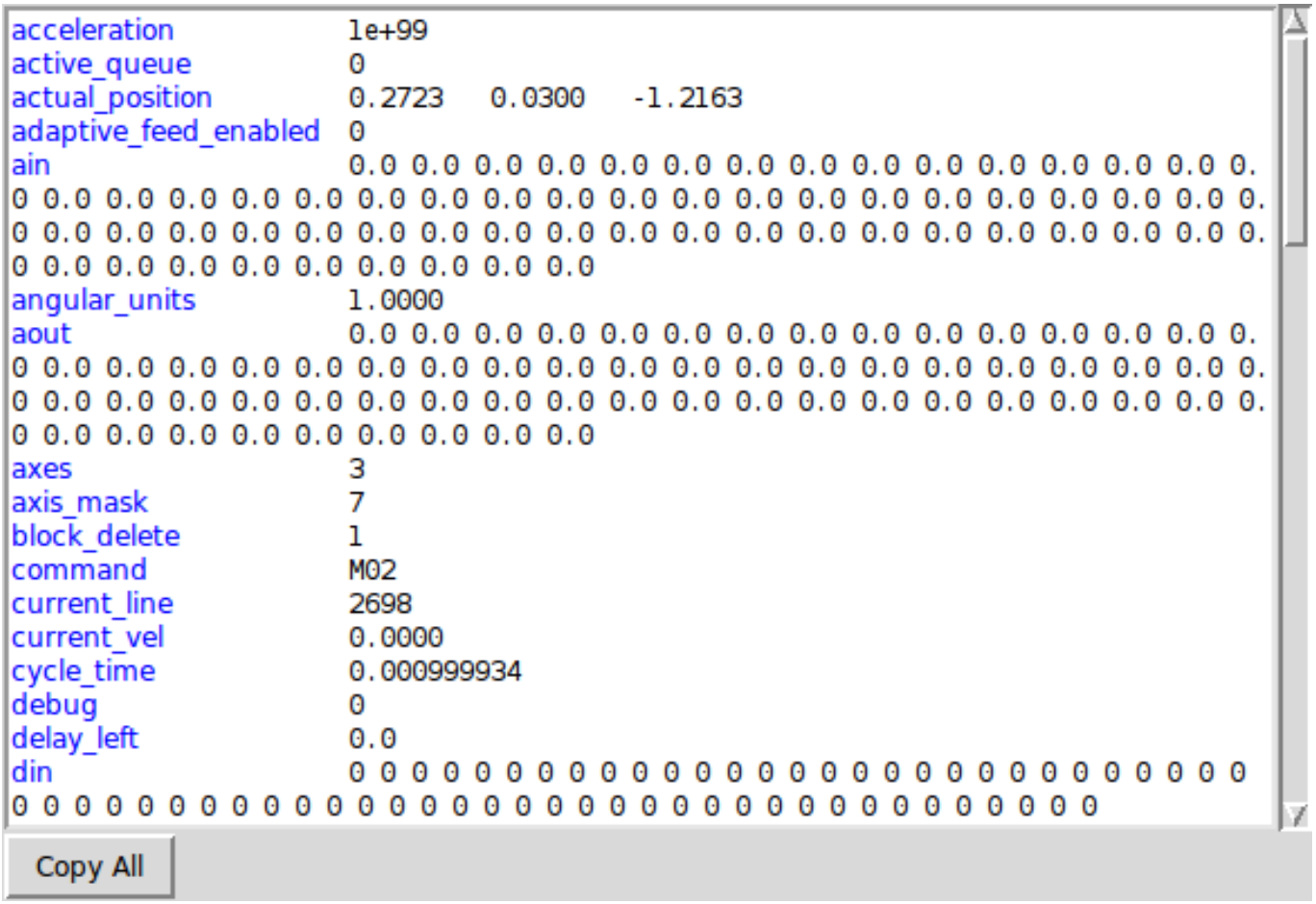



FIGURE 3.6 – Fenêtre d'état de LinuxCNC

Le nom de chaque entrée est affiché dans la colonne de gauche. La valeur courante de chaque entrée s'affiche dans la colonne de droite. Si la valeur a changé récemment, elle s'affiche en surbrillance rouge.

## 3.6 Entrée de données en texte (MDI)

AXIS inclut un programme appelé *mdi*, il permet d'envoyer des commandes à la session de LinuxCNC en cours, sous forme de lignes de texte. Vous pouvez lancer ce programme en ouvrant une console et en tapant:

```
mdi /chemin/vers/linuxcnc.nml
```

En cours d'exécution il affiche le prompt: *MDI>*. Quand une ligne vide est entrée, la position courante de la machine est affichée. Quand une commande est entrée, elle est passée à LinuxCNC qui l'exécute.

Voici un exemple de session MDI.

```
$ MDI ~/linuxcnc/configs/sim/emc.nml
MDI>
(0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
MDI> G1 F5 X1
MDI>
(0.5928500000000374, 0.0, 0.0, 0.0, 0.0, 0.0)
MDI>
(1.0000000000000639, 0.0, 0.0, 0.0, 0.0, 0.0)
```

## 3.7 axis-remote: Commande à distance de l'interface graphique d'AXIS

AXIS inclut un programme appelé *axis-remote* qui permet d'envoyer certaines commandes vers l'application AXIS fonctionnant à distance. Les commandes disponibles sont visibles en faisant: *axis-remote --help* pour vérifier qu'AXIS est en marche, inclure: (*--ping*), charger un fichier, recharger le fichier courant avec: (*--reload*) et quitter le programme AXIS avec: (*--quit*).

## 3.8 hal_manualtoolchange: Dialogue de changement d'outil manuel

LinuxCNC inclut un composant userspace HAL de appelé *hal_manualtoolchange*, il ouvre une fenêtre d'appel d'outil visible ci-dessous, quand la commande **M6** est invoquée. Dés que le bouton *Continuer* est pressé, l'exécution du programme reprend.

Le fichier de configuration HAL *configs/sim/axis_manualtoolchange.hal* montre les commandes HAL nécessaires pour l'utilisation de ce composant.

hal_manualtoolchange peut être utilisé même si l'interface graphique AXIS n'est pas en service. Cette composante est particulièrement utile si vous avez des outils de pré-réglage et que vous utilisez la table d'outils.

**Important**
Le parcours d'outil d'un mouvement en vitesse rapide ne sera pas visible s'il suit un changement d'outil T<n> et ce jusqu'au prochain mouvement en vitesse travail. Ceci peut être source de confusion pour l'opérateur. Pour contourner cette particularité, ajoutez toujours un G1 sans mouvement après un M6 T<n>.

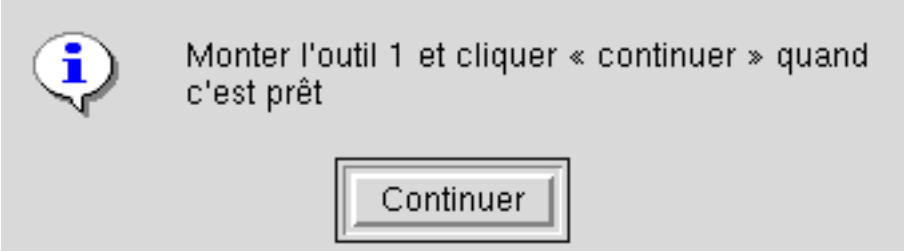



FIGURE 3.7 – La fenêtre de changement manuel d'outil

## 3.9 Modules en Python

AXIS inclut plusieurs modules en Python qui peuvent être très utiles. Pour des informations complètes sur ces modules, faites: *pydoc <nom du module* ou lisez son code source. Modules inclus:
– *LinuxCNC* fournit l'accès aux commandes de LinuxCNC, à son état et aux chaînes d'erreur
– *gcode* fournit l'accès à l'interpréteur RS274NGC
– *rs274* fournit des outils supplémentaires pour travailler sur les fichiers RS274NGC
– *hal* permet la création par l'utilisateur de composants de HAL écrits en Python
– *_togl* fournit des éléments OpenGL utilisables dans les applications Tkinter
– *minigl* fournit l'accès aux sous-ensembles d'OpenGL utilisés par AXIS
Pour utiliser ces modules dans vos propres scripts, assurez-vous que le répertoire où ils se trouvent est dans le chemin d'accès des modules Python. Avec une version installée de LinuxCNC, ça se fera automatiquement. Avec une version installée en *in-place*, ça peut être fait avec l'aide de: */scripts/rip-environment*.

## 3.10 Utiliser AXIS en mode tour

En incluant la ligne

```
LATHE = 1
```

dans la section [DISPLAY] du fichier ini, AXIS sélectionnera le mode tour. L'axe Y ne s'affiche pas parmi les coordonnées, la vue est modifiée pour placer la broche à gauche, l'axe Z horizontalement avec son côté positif vers la droite **(Z+)** et l'axe X s'étendant vers le bas de l'écran **(X+)**. Plusieurs contrôles (tels que ceux des vues prédéfinies) sont supprimés. Les lectures de coordonnées pour X sont désormais en diamètre et en rayon.

La touche **V** agit alors sur le zoom pour afficher le tracé complet du fichier chargé.

En mode tour (lathe), la forme et l'orientation de l'outil chargé sont représentés.

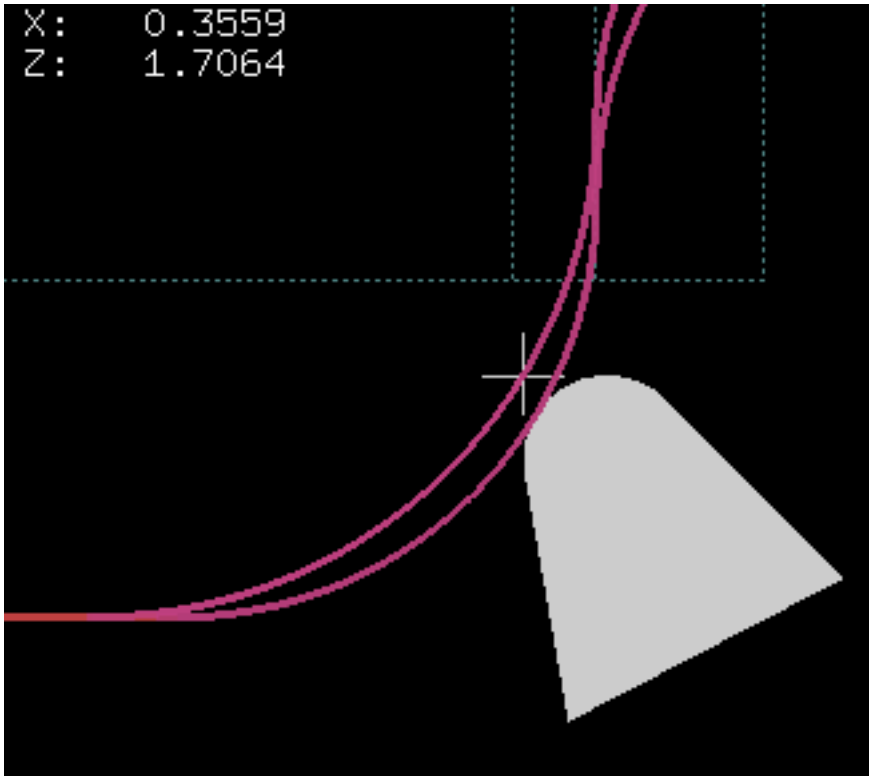



FIGURE 3.8 – Représentation de l'outil en mode tour

## 3.11 Configuration avancée d'AXIS

Pour plus d'informations sur les paramètres du fichier ini pouvant modifier le fonctionnement d'AXIS, voir le fichier ini, sections [DISPLAY] et le chapitre sur la configuration dans le manuel de l'intégrateur.

### 3.11.1 Filtres de programme

AXIS a la capacité d'envoyer des fichiers chargés à travers un *filtre de programme*. Ce filtre peut faire diverses tâches: Simple, comme s'assurer que le programme se termine bien par un ***M2*** ou complexe, comme détecter que l'entrée est une image et générer le g-code qui permettra d'usiner sa forme.

La section [FILTER] du fichier ini défini comment les filtres doivent agir. Premièrement, pour chaque type de fichier, écrire une ligne PROGRAM_EXTENSION puis, spécifier le programme à exécuter pour chaque type de fichier. Ce programme reçoit comme argument le nom du fichier d'entrée, il doit produire le G-code selon le standard rs274ngc, en sortie. Les lignes de cette sortie s'affichent alors dans la zone de texte, le parcours d'outil résultant est visible dans la zone graphique, enfin il sera exécuté quand LinuxCNC recevra la commande *Exécuter le programme*. Les lignes suivantes fournissent la possibilité d'utiliser *image-to-gcode*, le convertisseur d'images fourni avec LinuxCNC:

```
[FILTER]
PROGRAM_EXTENSION = .png,.gif Greyscale Depth Image
png = image-to-gcode
gif = image-to-gcode
```

Il est également possible de spécifier un interpréteur:

```
PROGRAM_EXTENSION = .py Python Script
py = python
```

De cette manière, n'importe quel script Python pourra être ouvert et sa sortie traitée comme du G-code. Un autre exemple est disponible dans: */nc_files/holecircle.py* . Ce script crée le G-code pour percer une série de trous suivant un arc de cercle.

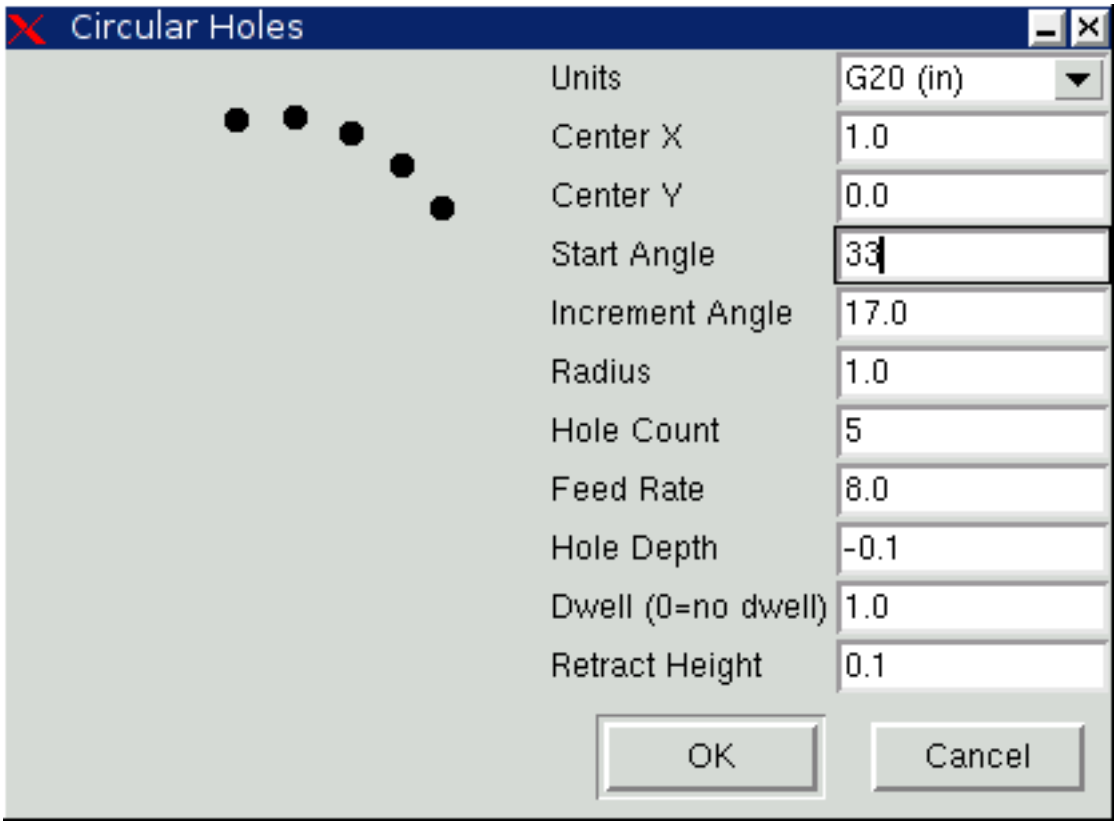



FIGURE 3.9 – Perçages circulaires

Si la variable d'environnement: AXIS_PROGRESS_BAR est active, alors les lignes seront écrites sur stderr de la forme:

```
FILTER_PROGRESS=%d
```

AXIS fixera la barre de progression selon le pourcentage donné. Cette fonction devrait être utilisée pour un filtre qui fonctionne suffisamment longtemps.

### 3.11.2 La base de données des ressources X

Les couleurs de la plupart des éléments de l'interface utilisateur d'AXIS peuvent être personnalisées grâce à la base de données X. Le fichier *axis_light_background* modifie les couleurs de la fenêtre du parcours d'outil sur le modèle *lignes noires et fond blanc*, il sert aussi de référence des éléments configurables dans l'écran graphique. L'exemple de fichier *axis_big_dro* évolution de la position de lecture à une police plus grande taille. Pour utiliser ces fichiers:

```
xrdb -merge /usr/share/doc/linuxcnc/axis_light_background

xrdb -merge /usr/share/doc/linuxcnc/axis_big_dro
```

Pour plus d'informations au sujet des éléments configurables dans les applications Tk, référez vous aux manuels de Tk.

Les bureaux graphiques modernes effectuent certains réglages dans la base de données des ressources X ces réglages peuvent affecter ceux d'AXIS, par défaut ces réglages sont ignorés. Pour que les éléments des ressources X écrasent ceux par défaut dans AXIS, il faut inclure cette ligne dans vos ressources X:

```
    *Axis*optionLevel: widgetDefault
```

ce qui entraînera la construction des options au niveau *widgetDefault*, de sorte que les ressources X (qui sont elles, au niveau *userDefault*) puissent l'emporter.

### 3.11.3 Manivelle de jog

Pour accroître l'interaction d'AXIS avec une manivelle de jog physique, l'axe actif courant sélectionné dans l'interface graphique est aussi reporté sur une *pin HAL* avec un nom comme *axisui.jog.x*. Excepté pendant un court instant après que l'axe courant ait changé, une seule de ces pins à la fois est *TRUE*, les autres restent *FALSE*.

Après qu'AXIS ait créé ces *HAL pins*, il exécute le fichier hal déclaré avec: [HAL]POSTGUI_HALFILE. Ce qui diffère de [HAL]HALFILE, qui lui ne s'utilise qu'une seule fois.

### 3.11.4 ~/.axisrc

Si il existe, le contenu de: *~/.axisrc* est exécuté comme un code source Python juste avant l'ouverture de l'interface graphique d'AXIS. Les détails de ce qui peut être écrit dans .axisrc sont sujets à changement durant le cycle de développement.

Les lignes visibles ci-dessous ajoutent un Ctrl+Q comme raccourci clavier pour Quitter et activer l'option *Distance restante* par défaut.

**Exemple de fichier .axisrc**

```
root_window.bind("<Control-q>", "destroy .")
help2.append(("Control-Q", "Quit"))
```

### 3.11.5 Éditeur externe

En définissant: [DISPLAY]EDITOR , les options de menu: *Fichier* → *Éditer* ainsi que *Fichier* → *Éditer la table d'outils*, deviennent accessibles. Deux valeurs qui fonctionnent bien: EDITOR=gedit et *EDITOR=gnome-terminal -e nano*.

### 3.11.6 Panneau de contrôle virtuel

AXIS peut afficher un panneau de commande virtuel personnalisé dans le volet de droite. Il est possible d'y placer des boutons, des indicateurs qui afficheront des données et plus encore. Voir le manuel de l'intégrateur.

### 3.11.7 Commentaires spéciaux

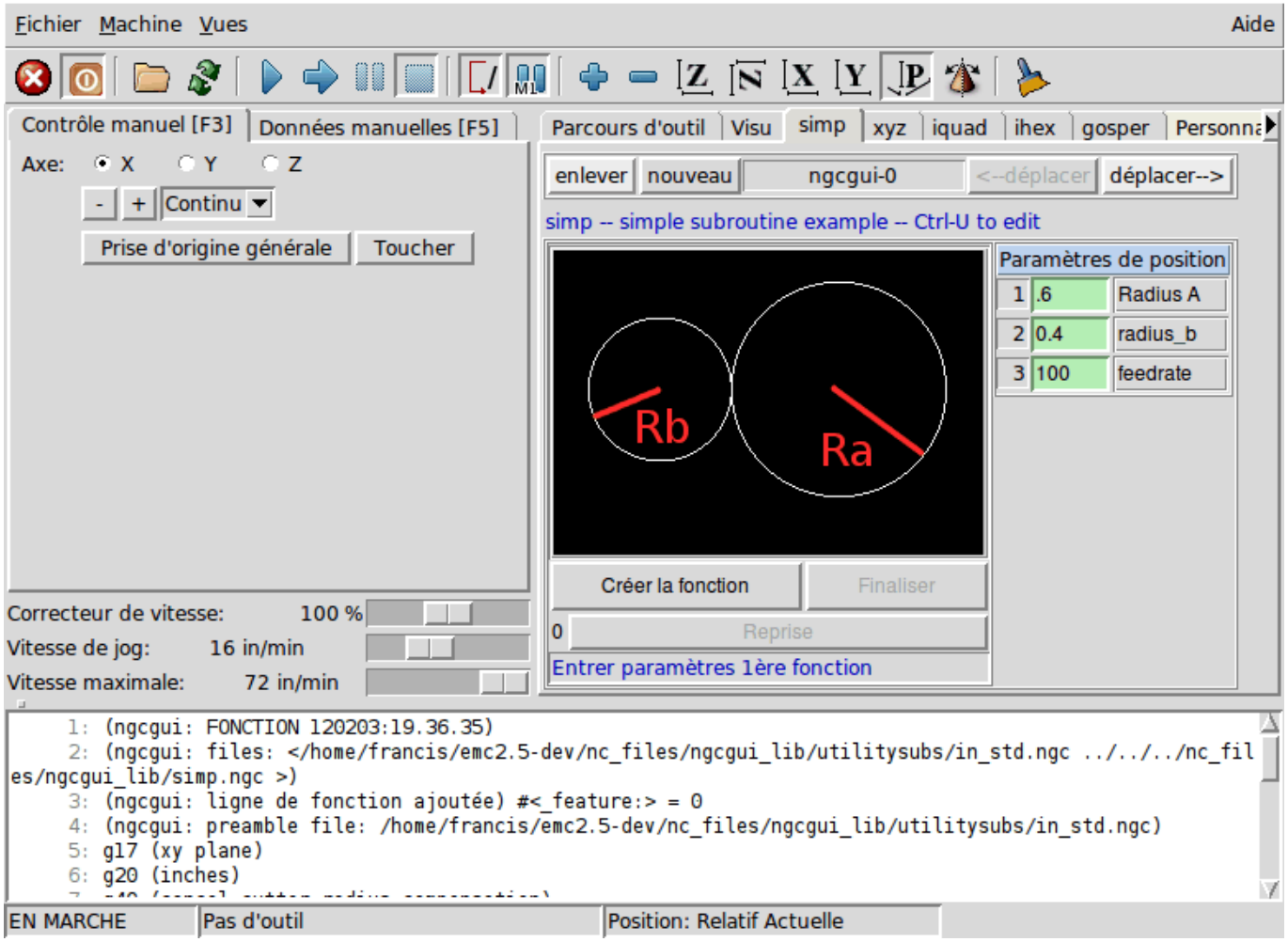
Les commentaires spéciaux peuvent être insérés dans le fichier de G-code pour contrôler le comportement de l'affichage d'AXIS. Pour limiter l'aperçu au seul affichage du parcours d'outil, utiliser ces commentaires spéciaux. Rien ne s'affichera entre le commentaire (AXIS,hide) et le commentaire (AXIS,show) sauf le parcours d'outil. Les (AXIS,hide) et (AXIS,show) doivent être utilisés par paires avec le (AXIS, hide) en premier. Tout ce qui est après un (AXIS,stop) ne sera pas visible.

Ces commentaires sont utiles pour désencombrer l'affichage d'aperçu (Par exemple pendant le débogage d'un grand fichier G-code, on peut désactiver l' aperçu sur certaines parties qui sont déjà fonctionnelles).

- (AXIS,hide) Arrête le parcours d'outil (à placer en premier)
- (AXIS,show) Reprend le parcours d'outil (il faut suivre un cache)
- (AXIS,stop) Arrête le parcours d'outil d'ici à la fin du fichier.
- (AXIS,notify,le_texte) Affiche *le_texte* à l'écran, comme une info. Cet affichage peut être très utile lors du pré-affichage du parcours d'outil, quand les commentaires (debug,message) ne sont pas affichés.

# Chapitre 4

# L'utilitaire graphique NGCGUI




## 4.1 Vue d'ensemble

- NGCGUI est un utilitaire pour écrire et utiliser les sous-programmes avec LinuxCNC.
- NGCGUI peut être utilisé en autonome ou intégré, dans ce dernier cas, il crée des onglets multiples sur la page de l'interface graphique Axis.

Ngcgui est un outil puissant pour construire les programmes de G-code en sous-programmes.

- Les sous-programmes peuvent être concaténés pour fournir un programme de G-code complet.
- De multiples instances d'un sous-programme peuvent être utilisées pour fournir la même tâche à différents emplacements sur la même pièce.
- N'importe quel G-code valide peut être utilisé dans un sous-programme.
- De nouveaux sous-programmes peuvent être ajoutés à la volée.

## 4.2 Configurations fournies en exemple.

Trois configurations sont fournies, elles se trouve dans le répertoire *sim* du sélecteur de configuration de LinuxCNC. Le sélecteur de configuration se trouve quand à lui dans le menu *Applications* → *CNC* → *LinuxCNC*.
- *ngcgui* - Un exemple facile à comprendre utilisant ces sous-programmes:
  - *simp* - Un exemple simple créant deux cercles.
  - *xyz* - Crée une boîte basée sur deux coins opposés.
  - *iquad* - Crée un quadrilatère interne.
  - *db25* - Crée la découpe pour une fiche DB25.
  - *ihex* - Crée un hexagone interne.
  - *gosper* - Une démo sur la récursion.
  - *Custom* - Crée des onglets personnalisés.
  - *ttt* - Traceur True Type, pour créer des textes à graver.
- *ngcgui-lathe* - Un exemple de sous-programme pour un tour:
  - *id* - Alésage intérieur.
  - *od* - Cylindrage extérieur.
  - *taper-od* - Tourne un cône mâle.
  - *Custom* - Crée des onglets personnalisés.
- *ngcgui-simple* - Un exemple simple:
  - *simp* - Un exemple simple créant deux cercles.
  - *xyz* - Crée une boîte basée sur deux coins opposés.

Pour visualiser les sous-programmes presser l'***A**rrêt d'**U**rgence* puis, activer la *Marche Machine* et réaliser la *Prise d'origine générale*. Cliquer sur un onglet de ngcgui et presser *Créer la fonction* puis *Finaliser*. Enfin, presser sur le bouton *Départ cycle* pour exécuter le G-code.

**Note**
Les sous-programmes d'exemples fournis avec la distribution doivent tous fonctionner avec la configuration de la machine simulée. Un utilisateur doit toujours comprendre le comportement et les implications d'un programme avant de tenter de l'exécuter sur une machine réelle.

## 4.3 Librairies

Les configurations en simulation pour ngcgui utilisent les liens suivants vers des librairies de LinuxCNC protégées en écriture:
- *Sous-fichiers compatibles ngcgui* - ngcgui_lib
- *Sous-programme d'aide* - ngcgui_lib/utilitysubs
- *Fichiers M utilisateurs* - ngcgui_lib/mfiles

Ces librairies sont définies dans le fichier ini par les items:

```
[RS274NGC]
SUBROUTINE_PATH = ../../../nc_files/ngcgui_lib:../../../nc_files/ngcgui_lib/utilitysubs
USER_M_PATH     = ../../../nc_files/ngcgui_lib/mfiles
```

C'est une longue ligne (ne pas continuer sur de multiples lignes) qui spécifie les répertoires utilisés dans le chemin de recherche. Les noms de répertoires sont séparés par le caractère (:).

L'utilisateur peut créer de nouveaux répertoires pour ses propres sous-programmes et fichiers M et les ajouter dans le chemin de recherche.

Par exemple, un utilisateur pourrait créer ces répertoires à partir de la console.

```
mkdir /home/myusername/mysubs
mkdir /home/myusername/mymfiles
```

Puis y créer ou y copier des fichiers qui lui seront accessibles en écriture. Par exemple, créer un sous-fichier compatible ngcgui nommé:

```
/home/myusername/mysubs/exemple.ngc
```

Le fichier ini doit être édité pour lui inclure les nouveaux sous-fichiers et les ajouter au chemin. Pour cet exemple:

```
[RS274NGC]
SUBROUTINE_PATH = /home/myusername/mysubs:../../../nc_files/ngcgui_lib:../../../nc_files/ ↩
    ngcgui_lib/utilitysubs
USER_M_PATH     = /home/myusername/mymfiles:../../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
NGCGUI_SUBFILE = exemple.ngc
```

LinuxCNC et ngcgui utilisent le premier fichier trouvé lors d'une recherche dans les répertoires du chemin de recherche. Avec ce comportement, Il est possible de substituer un sous-fichier ngcgui_lib en plaçant un sous-fichier avec un nom identique plus tôt dans le chemin de recherche pour qu'il soit trouvé avant. Plus d'informations peuvent être trouvées au chapitre INI dans le Manuel de l'intégrateur.

## 4.4 Intégration de ngcgui dans Axis

D'autres exemples de sous-programmes se trouvent dans le répertoire sim/ngcgui

Les items de fichier INI pour NGCGUI vont dans la section [DISPLAY].
- *TKPKG* = Ngcgui 1.0 - Le paquet principal de NGCGUI (doit précéder Ngcguittt).
- *TKPKG* = Ngcguittt 1.0 - Le paquet du traceur True Type pour générer des textes à graver.
- *NGCGUI_FONT* = Helvetica -12 normal - Spécifie la police utilisée.
- *NGCGUI_PREAMBLE* = in_std.ngc - Le fichier de préambule à ajouter au début du sous-programme. Quand plusieurs sous-programmes sont concaténés, un seul est ajouté.
- *NGCGUI_SUBFILE* = simp.ngc - Crée un onglet depuis le sous-programme nommé.
- *NGCGUI_SUBFILE* = "" - Crée un onglet personnalisé.
- *#NGCGUI_OPTIONS* = opt1 opt2 . . . - Options Ngcgui
  - # opt items:
    - # nonew — interdit la création d'un nouvel onglet personnalisé
    - # noremove — interdit l'effacement d'une page d'onglet
    - # noauto — pas d'envoi auto (makeFile, puis envoi manuel)
    - # noiframe — no internal image, image on separate top level
- *TTT* = Le programme True-type Tracer
- *TTT_PREAMBLE* = in_std.ngc - Optionnel, spécifie le nom de fichier de préambule utilisé par ttt pour créer les sous-fichiers.

Voici un exemple d'intégration de NGCGUI dans Axis. Les sous-programmes doivent être placés dans un répertoire spécifié par la variable [RS274NGC]SUBROUTINE_PATH. Certains exemples de sous-programmes utilisent d'autres sous-programmes, bien vérifier pour être sur d'avoir les bonnes dépendances, le cas échéant, dans un répertoire SUBROUTINE_PATH. Certains sous-programmes peuvent utiliser des fichiers M (Mfiles) personnalisés qui doivent se trouver dans un répertoire spécifié par [RS274NGC]USER_M_PATH.

**Note**
Il ne s'agit pas d'un fichier ini complet, les items montrés sont ceux utilisés par ngcgui. D'autres items sont requis par LinuxCNC pour obtenir un fichier ini complet.

**Simple fichier.ini**

```
[RS274NGC]
SUBROUTINE_PATH   = ../../../nc_files/ngcgui_lib:../../../ngcgui_lib/utilitysubs
USER_M_PATH       = ../../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
TKPKG             = Ngcgui    1.0
TKPKG             = Ngcguittt 1.0
# Ngcgui must precede Ngcguittt

NGCGUI_FONT       = Helvetica -12 normal
# specifie seulement les noms de fichiers, doit être dans [RS274NGC]SUBROUTINE_PATH
NGCGUI_PREAMBLE   = in_std.ngc
NGCGUI_SUBFILE    = simp.ngc
NGCGUI_SUBFILE    = xyz.ngc
NGCGUI_SUBFILE    = iquad.ngc
NGCGUI_SUBFILE    = db25.ngc
NGCGUI_SUBFILE    = ihex.ngc
NGCGUI_SUBFILE    = gosper.ngc
# specifie "" pour une page d'onglet épersonnalise
NGCGUI_SUBFILE    = ""
#NGCGUI_SUBFILE   = "" éutilis quand une trame d'image est éespcifie si
#                      ouvrir d'autres fichiers est requis
#                      les images seront mises dans une êfentre de haut niveau
NGCGUI_OPTIONS    =
#NGCGUI_OPTIONS   = opt1 opt2 ...
# opt items:
#   nonew      -- interdit la écration d'un nouvel onglet épersonnalis
#   noremove   -- interdit l'effacement d'une page d'onglet
#   noauto     -- pas d'envoi auto (makeFile, puis envoi manuel)
#   noiframe   -- no internal image, image on separate top level

TTT               = truetype-tracer
TTT_PREAMBLE      = in_std.ngc

PROGRAM_PREFIX    = ../../nc_files
```

### 4.4.1 Traceur Truetype

Ngcgui_ttt fourni le support pour truetype-tracer (v4). Il crée un onglet sur Axis qui permet à l'utilisateur de créer ses propres textes dans de nouveaux onglets ngcgui et en choisissant leurs fontes et autres paramètres. (Truetype-tracer doit être installé indépendamment).

L'intégration de ngcgui_ttt dans Axis, nécessite les items suivants en plus de ceux de ngcgui:

```
Item:    [DISPLAY]TKPKG = Ngcgui_ttt numéro_de_version
Exemple: [DISPLAY]TKPKG = Ngcgui_ttt 1.0
Note:    Obligatoire, spécifie le chargement de ngcgui_ttt dans un onglet
d'Axis nommé ttt. Doit suivre l'item TKPKG = Ngcgui.

Item:    [DISPLAY]TTT = chemein_de_truetype-tracer
Exemple: [DISPLAY]TTT = truetype-tracer
Note:    Optionnel, s'il n'est pas spécifié, utilisera
/usr/local/bin/truetype-tracer. Spécifier avec un chemin absolut ou simplement
le nom de l'exécutable, dans ce cas, la variable d'environnement PATH de
l'utilisateur sera utilisée pour rechercher le programme.

Item:    [DISPLAY]TTT_PREAMBLE = nom_fichier_préambule
```

```
Exemple: [DISPLAY]TTT_PREAMBLE = in_std.ngc
Note:    Optionnel, spécifie le nom du fichier de préambule utilisé pour
les sous-fichiers créés par ttt.
```

### 4.4.2 Exemples d'INI

Ngcgui utilise le chemin de recherche de LinuxCNC pour chercher les fichiers.

Le chemin de recherche commence avec le répertoire standard spécifié par:

```
[DISPLAY]PROGRAM_PREFIX
```

suivi par les répertoires multiples spécifiés par:

```
[RS274NGC]SUBROUTINE_PATH
```

**Répertoires** Les répertoires peuvent être spécifiés comme des chemins absolus ou des chemins relatifs.

```
Exemple: [DISPLAY]PROGRAM_PREFIX = /home/myname/linuxcnc/nc_files
Exemple: [DISPLAY]PROGRAM_PREFIX = ~/linuxcnc/nc_files
Exemple: [DISPLAY]PROGRAM_PREFIX = ../../../nc_files
```

**Chemins absolus** Un chemin absolu commence avec un "/" qui indique un emplacement par rapport au système de fichiers complet. Un chemin qui commence par "~/" indique un chemin commençant *depuis* le répertoire home de l'utilisateur. Un chemin qui commence par "~nomutilisateur/" indique un chemin commençant *dans* le répertoire utilisateur.

**Chemins relatifs** Un chemin relatif commence dans le répertoire de démarrage qui est celui contenant le fichier ini. L'usage des chemins relatifs facilite l'accès aux configurations mais requiert une bonne compréhension de la façon dont les chemins sont spécifiés sous Linux.

```
./d0      est le même que d0, ex: un répertoire nommé d0 dans le répertoire de
départ.
../d1     se réfère au répertoire d1 dans le répertoire parent.
../../d2  se réfère au répertoire d2 dans le répertoire parent du parent.
../../../d3 etc.
```

Des répertoires multiples peuvent être spécifiés par la variable: [RS274NGC]SUBROUTINE_PATH suivie des chemins séparés par le signe ":". L'exemple suivant illustre le format utilisé pour les chemins multiples et montre l'utilisation de répertoires relatifs et absolus.

Exemple: *[RS274NGC]SUBROUTINE_PATH = ../../../nc_files/ngcgui_lib: ../../../nc_files/ngcgui_lib/utilitysubs:/tmp/tmpngc*

C'est une longue ligne, ne pas continuer sur de multiples lignes. Quand LinuxCNC et/ou Ngcgui cherchent un fichier, c'est le premier trouvé qui est utilisé.

LinuxCNC (et NGCGUI) doivent pouvoir trouver tous les sous-programmes avec les routines additionnelles qui sont appelées depuis les sous-fichiers NGCGUI. Il est pratique de placer les fichiers utilitaires dans un répertoire séparé comme indiqué dans l'exemple précédent.

La distribution inclus le répertoire ngcgui_lib et les fichiers de préambule , sous-fichiers, postambule et fichiers d'aide pour les démos. Pour modifier le comportement des fichiers, il est possible de copier n'importe quel fichier et de le placer en avant du chemin de recherche. Le premier répertoire recherché est: [DISPLAY]PROGRAM_PREFIX. Il est possible de l'utiliser mais c'est une meilleure pratique de créer un répertoire dédié en le plaçant au début du chemin donné par [RS274NGC]SUBROUTINE_PATH.

Dans l'exemple suivant, les fichiers dans /home/myname/emc2/mysubs seront trouvés avant ceux étant dans ../../../nc_files/ngcgui_lib.

Exemple: *[RS274NGC]SUBROUTINE_PATH = /home/myname/emc2/mysubs:../../../nc_files/ ngcgui_lib:../../../nc_files/ngcgui_lib/utilit*

Les débutants pourraient essayer par inadvertance d'utiliser des fichiers non structurés comme le nécessite ngcgui. Ngcgui déclencherait alors rapidement de nombreuses erreurs si les fichiers ne répondent pas à ses conventions. Une bonne pratique suggère que les sous-fichiers compatibles ngcgui doivent être placés dans un répertoire dédié à cette fin et que les préambules, postambules et fichiers d'aide doivent être dans un répertoire séparés pour dissuader toute tentative d'utilisation de ces sous-fichiers.

Pour intégrer ngcgui dans Axis, spécifier les items suivants dans le fichier ini:

```
Item:    [DISPLAY]PROGRAM_PREFIX = dirname
Exemple: [DISPLAY]PROGRAM_PREFIX = ../../../nc_files
Note:    Obligatoire et nécessaire pour de nombreuses fonctions de LinuxCNC.
         C'est le premier répertoire utilisé lors de la recherche de fichiers.

Item:    [RS274NGC]SUBROUTINE_PATH = dirname1:dirname2:dirname3 ...
Exemple: [RS274NGC]SUBROUTINE_PATH = ../../../nc_files/ngcgui_lib:../../../nc_files/ngcgui
Note:    Optionnel, mais très utile pour organiser les sous-fichiers et les
         fichiers utilitaires.

Item:    [DISPLAY]TKPKG=Ngcgui version_number
Exemple: [DISPLAY]TKPKG=Ngcgui 1.0
Note:    Obligatoire, spécifie le chargement des onglets ngcgui dans axis.

Item:    [DISPLAY]NGCGUI_FONT = font_descriptor
Exemple: [DISPLAY]NGCGUI_FONT = Helvetica -12 normal
Note:    Optionnel, descripteur de fontes compatible avec celui de Tcl.
         Avec des items pour le type de fonte -fontsize fontweight
         Par défaut c'est la police: Helvetica -10 normal

Item:    [DISPLAY]NGCGUI_SUBFILE = subfile_filename
Exemple: [DISPLAY]NGCGUI_SUBFILE = simp.ngc
Exemple: [DISPLAY]NGCGUI_SUBFILE = xyz.ngc
Exemple: [DISPLAY]NGCGUI_SUBFILE = ""
Note:    Utilise un ou plusieurs items pour spécifier les fichiers compatibles
         avec les sous-fichiers ngcgui qui requiert un onglet dans Axis au
         départ. Un onglet "personnalisé" est créé quand le nom de fichier est "".
         Un utilisateur peut utiliser l'onglet "Personnalisé" pour lire un
         fichier système et identifier un préambule, un sous-fichier ou un
         postambule.

Item:    [DISPLAY]NGCGUI_PREAMBLE = preamble_filename
Exemple: [DISPLAY]NGCGUI_PREAMBLE = in_std.ngc
Note:    Optionnel, si spécifié, alors ce fichier sera prépondérant à tous les
         sous-fichiers. Les fichiers créés avec l'onglet "Personnalisé"
         utilisent le préambule spécifié avec cette page.

Item:    [DISPLAY]NGCGUI_POSTAMBLE = postamble_filename
Exemple: [DISPLAY]NGCGUI_POSTAMBLE = bye.ngc
Note:    Optionnel, si spécifié, le fichier est ajouté à tous les sous-fichiers.
         Les fichiers créés avec l'onglet "Personnalisé" utilisent le postambule
         spécifié avec cette page.

Item:    [DISPLAY]NGCGUI_OPTIONS = opt1 opt2 ...
Exemple: [DISPLAY]NGCGUI_OPTIONS = nonew noremove
Note:    De multiples options séparées par des blancs.
         Par défaut,  ngcgui gère les onglets de cette manière:
            1) Un utilisateur peut créer de nouveaux onglets.
            2) Un utilisateur peut enlever des onglets (excepté le dernier
         restant)
```

```
   3) La finalisation envoie automatiquement les fichiers à Axis.
   4) Une trame d'image (iframe) est rendue disponible pour afficher
   une image pour le sous-fichier.

Les options _nonew_, _noremove_, _noauto_, _noiframe_ respectivement,
désactivent ces comportements par défaut.

Par défaut, Si un fichier d'image (.png, .gif, .jpg, .pgm)
est trouvé dans le même répertoire que le sous-fichier, l'image
est affichée dans une iframe.  Spécifier l'option
_noiframe_ rendra disponibles d'autres boutons pour sélectionner
un préambule, un sous-fichier ou un postambule et
des cases à cocher additionnelles. Les cases à cocher sont toujours
disponibles avec les touches spéciales suivantes:
  Ctrl-R Bascule "Conserver les valeurs à la lecture du sous-fichier"
  Ctrl-E Bascule "Déployer le sous-programme"
  Ctrl-a Bascule "EnvoiAuto"
 (Ctrl-k lists all keys and functions)

Si _noiframe_ est spécifié et qu'un fichier image est trouvé,
l'image est affichée dans une fenêtre séparée et
toutes les fonctions sont disponibles dans la page de l'onglet.

Les _NGCGUI_OPTIONS_ s'appliquent à tous les onglets ngcgui excepté
ceux sur lesquels les options _nonew_, _noremove_, et _noiframe_ ne
sont pas applicables pour l'onglet "Personnalisé".
Ne pas utiliser l'onglet "Personnalisé" si les utilisateurs doivent
avoir des possibilités de sélection de sous-fichiers et de création
d'onglet additionnels limitées.
```

## 4.5 Besoins des sous-programmes

Un sous-fichiers compatible NGCGUI contient une simple définition de sous-programme. Le nom du sous-programme doit être le même que celui du fichier (non inclus l'extension .ngc). LinuxCNC supporte les sous-programmes nommés ou numérotés, mais seuls les sous-programmes nommés sont compatible avec NGCGUI. Pour plus d'informations voir le chapitre sur les O-Codes.

La première ligne, autre qu'un commentaire, doit être une déclaration *sub*. La dernière ligne, autre qu'un commentaire, doit être une déclaration *endsub*.

**exemple.ngc:**

```
o<exemple> sub
  CORPS DU SOUS-PROGRAMME
o<exemple> endsub
```

Le corps du sous-programme doit commencer par un jeu de déclarations définissant les paramètres nommés locaux pour chaque paramètre positionnel attendu pour l'appel du sous-programme. Ces définitions doivent être consécutives, commencer par #1 et finir avec le numéro du dernier paramètre utilisé. Les définitions doivent être fournies pour chacun de ces paramètres (aucune omissions).

**Numérotation des paramètres**

```
#<xparm> = #1
#<yparm> = #2
#<zparm> = #3
```

LinuxCNC considère tous les paramètres numérotés entre #1 est #30 comme étant des paramètres appelables, de même, ngcgui fourni des dialogues de saisie pour n'importe quel paramètres dans cette fourchette. Il est de bonne pratique d'éviter d'utiliser

un paramètre numéroté de #1 jusqu'à #30 n'importe où ailleurs dans le sous-programme. L'utilisation de paramètres nommés locaux est recommandée pour toutes le variables internes.

Chaque définition de déclaration peut optionnellement inclure un commentaire spécial et une valeur par défaut pour le paramètre.

**Prototypage de déclaration**

```
#<vname> = #n (=évaleur_par_dfaut)
ou
#<vname> = #n (texte_de_commentaire)
ou
#<vname> = #n (=évaleur_par_dfaut texte_de_commentaire)
```

**Exemples de paramètres**

```
#<xparm> = #1 (=0.0)
#<yparm> = #2 (Ystart)
#<zparm> = #3 (=0.0 Z start setting)
```

Si une valeur_par_défaut est donnée, elle sera placée au démarrage, dans la boîte de saisie pour le paramètre.

Si un texte_de_commentaire est inclus, il sera utilisé pour identifier l'entrée à la place du nom du paramètre.

**Paramètres nommés globaux** Note sur les paramètres nommés globaux (#<nom_global>) avec ngcgui:

Comme dans de nombreux langages de programmation, l'utilisation de variables globales est puissante, mais peut souvent mener à des conséquences inattendues. Dans LinuxCNC, les paramètres nommés globaux existants seront valides lors de l'exécution du sous-programme et les sous-programmes peuvent les modifier ou en créer.

L'utilisation de paramètres nommé globaux comme entrées dans un sous-programme est déconseillé parce-que de tels usages requiert l'établissement et la maintenance d'un contexte global bien défini, ce qui est problématique à maintenir. L'utilisation de paramètres numérotés en #1 et #30 devrait être suffisant pour satisfaire les besoins les plus exigeants.

Ngcgui supporte quelques entrées par paramètres nommés globaux mais leurs usage est obsolète et non documenté ici.

Bien que les entrées par paramètres nommés globaux soient déconseillées, les sous-programmes LinuxCNC doivent utiliser des paramètres nommés globaux pour retourner les résultats. Puisque les sous-fichiers compatibles ngcgui sont destinés à l'usage de l'interface graphique, les valeurs de retour n'ont pas d'exigence commune. Toutefois, ngcgui est utile comme outil de test pour les sous-programmes qui ne retournent pas de paramètres nommés globaux et il est commun pour les sous-fichiers compatibles ngcgui d'appeler des fichiers de sous-programmes utilitaires qui eux retournent des résultats avec des paramètres nommés globaux.

Pour supporter ces usages, ngcgui ignore les paramètres nommés globaux qui incluent le caractère (:) dans leur nom. Utilisation des deux points (:) dans un nom prévient ngcgui de créer une bîte de saisie pour ces paramètres.

**Paramètres nommés globaux**

```
o<exemp> sub
...
#<_exemp:result> = #5410       (retourne le èdiamtre de l'outil courant)
...
o<helper> call [#<x1>] [#<x2>] (appel d'un sous-programme)
#<xresult> = #<_helper:answer> (localise éimmdiatement le érsultat du
fichier d'aide)
#<_helper:answer> = 0.0 (rend nul le èparamtre énomm global éutilis par le
sous-programme)
...
o<exemp> endsub
```

Dans l'exemple précédent, le sous-programme utilitaire sera trouvé dans un fichier séparé nommé helper.ngc. Le sous-programme d'aide retourne un résultat dans un paramètre nommé global nommé #<_helper:answer>.

Pour une bonne pratique, le sous-fichier appelant localise immédiatement le résultat pour une utilisation ailleurs dans le sous-fichier et le paramètre nommé global, utilisé pour retourner le résultat est mis à zéro pour diminuer les chances qu'il soit utilisé par inadvertance ailleurs dans le contexte global. (La mise à zéro avec 0.0 n'est pas toujours le meilleur choix).

Ngcgui supporte la création et la concaténation de multiples fonctions pour un sous-fichier et pour de multiples sous-fichiers. Il est parfois pratique pour les sous-fichiers de déterminer leur ordre au début de l'exécution afin que ngcgui insère un paramètre global spécial qui pourra être testé par tous les sous-programmes. Ce paramètre est nommé #<_feature:>. Sa valeur commence avec 0 et est incrémentée avec chaque fonction qui lui est ajoutée.

**Fonctions additionelles** Un commentaire spécial *info* peut être inclus quelque part dans les sous-fichier compatibles ngcgui. Le format est le suivant:

```
(info: info_text)
```

La chaine *info_text* est affichée vers le haut de la page de l'onglet ngcgui dans Axis.

Les fichiers non destinés à servir de sous-fichiers peuvent inclure le commentaire spécial: "(not_a_subfile)" de sorte que ngcgui les rejette automatiquement avec un message explicatif.

```
(not_a_subfile)
```

Un fichier image optionnel (.png, .gif, .jpg, .pgm) peut accompagner un sous-fichier. Le fichier image peut aider à clarifier les paramètres utilisés par le sous-fichier. Le fichier image doit être dans le même répertoire que le sous-fichier et doit avoir le même nom avec une extension appropriée au fichier image, ex: le sous-fichier exemp.ngc doit être accompagné par l'image exemp.png. Ngcgui tente de redimensionner de grandes images par sous-échantillonnage à une largeur maximale de 320 et une hauteur maximum de 240 pixels.

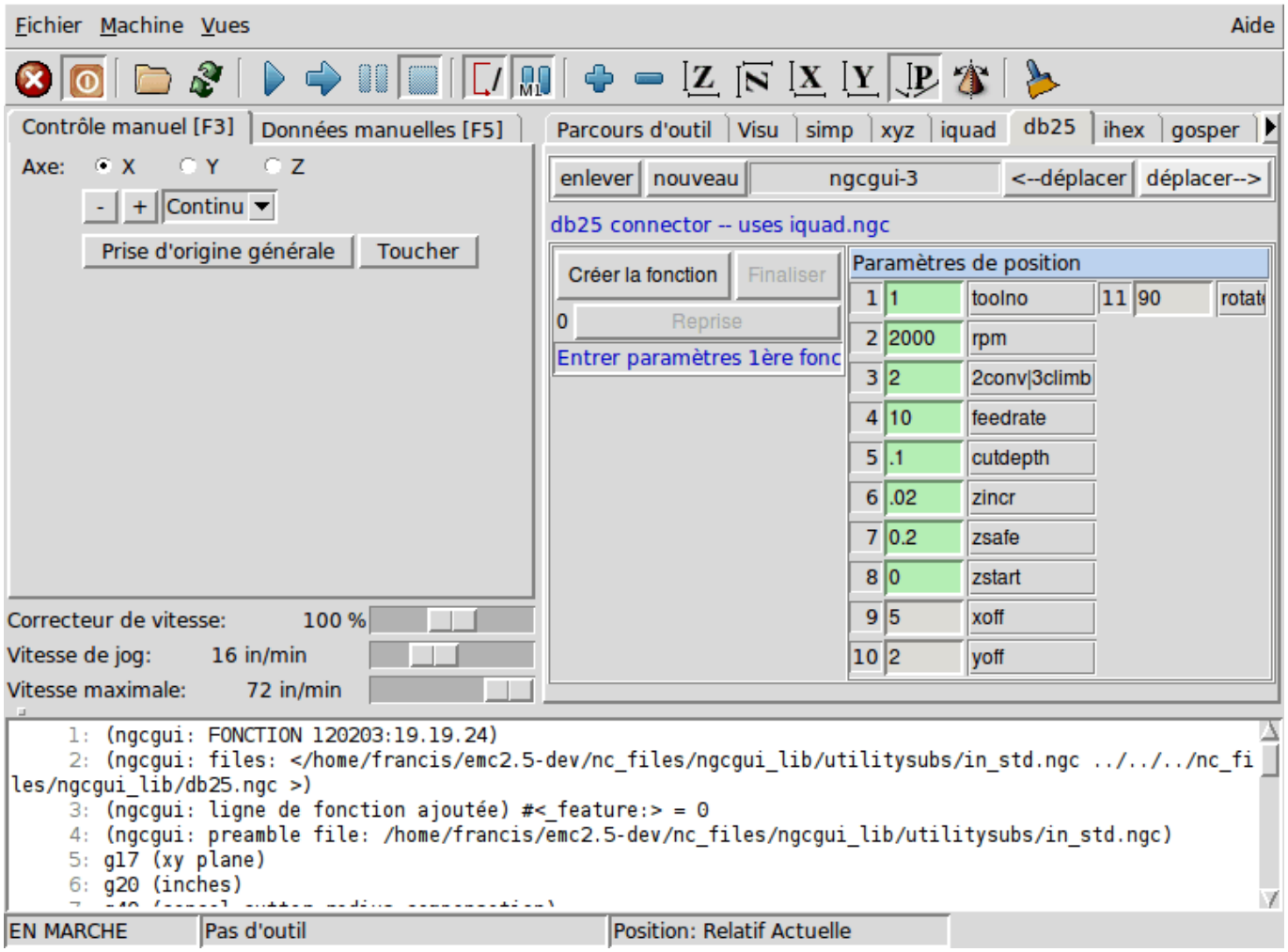
Aucune des conventions nécessaires pour faire une sous-fichier compatible ngcgui n'empêche son utilisation en tant que fichier de sous-programme pour LinuxCNC.

La distribution LinuxCNC inclus une librairie (répertoire ngcgui_lib) qui contient plusieurs exemples de sous-fichiers et de fichiers utilitaires compatibles ngcgui pour illustrer les fonctions des sous-programmes de LinuxCNC et l'usage de ngcgui.

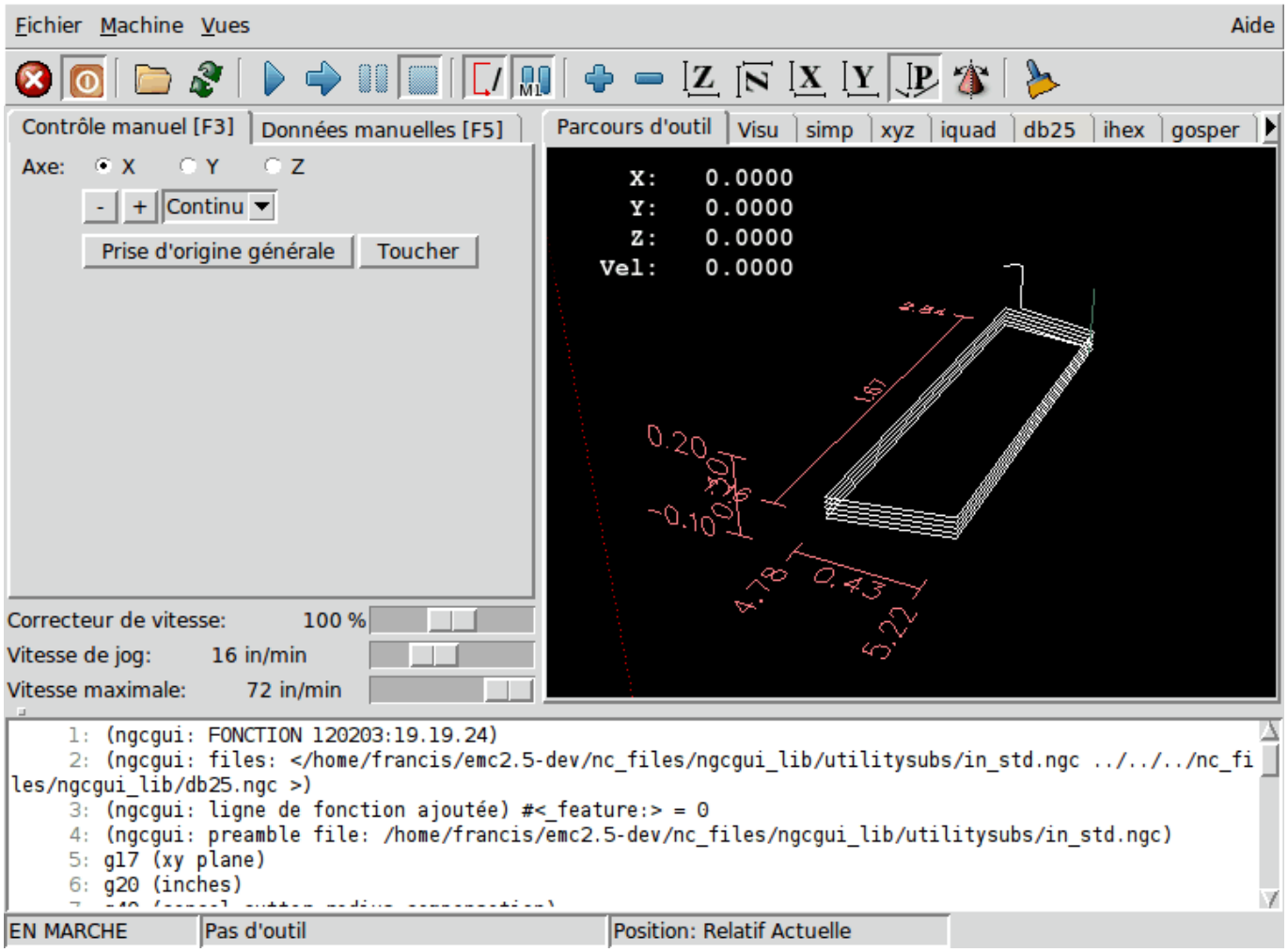
Des sous-programmes additionnels soumis par les utilisateurs se trouvent dans le forum dans la section *Subroutines*.

## 4.6 Exemple, découpe pour DB25

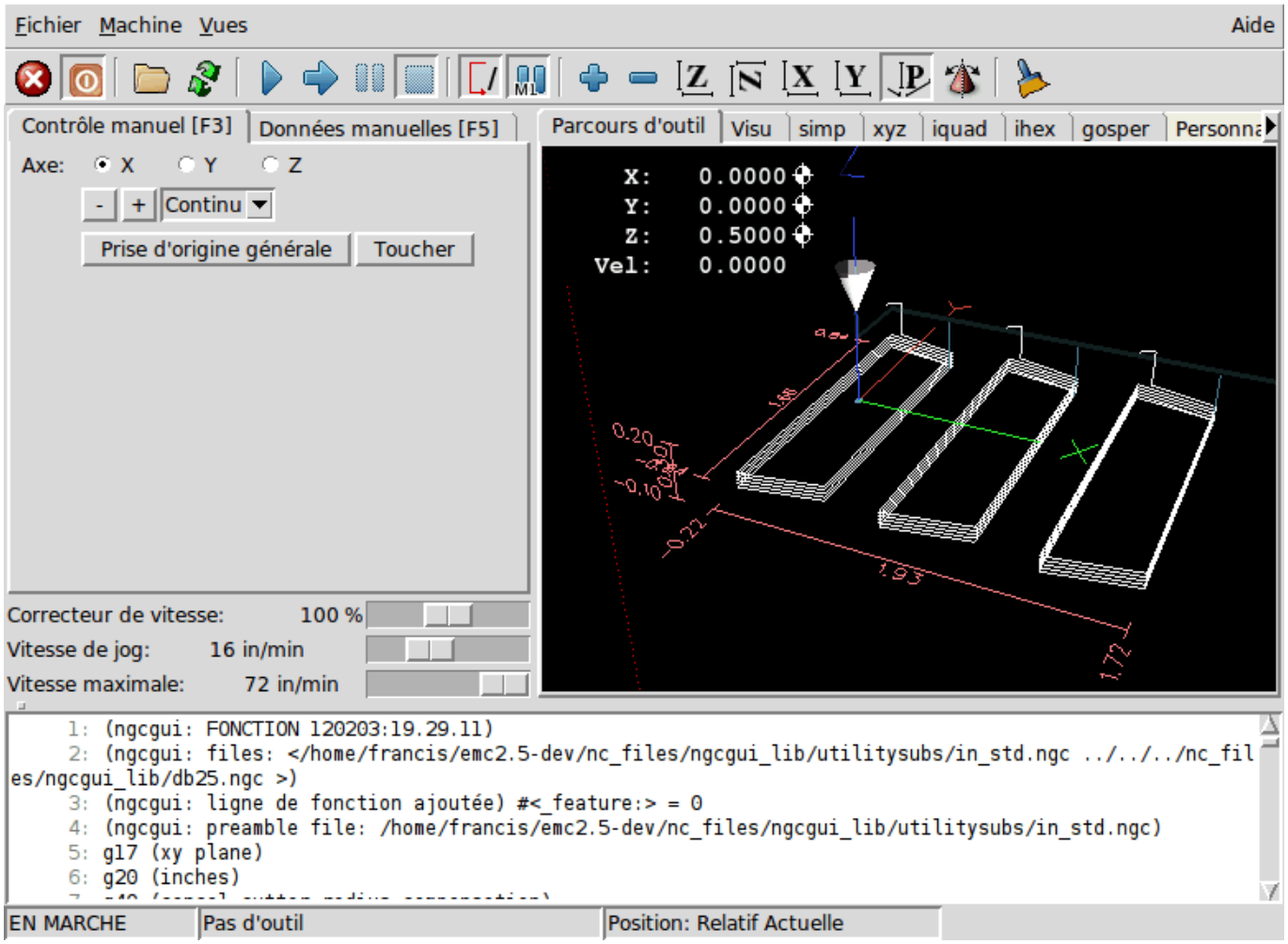
L'exemple ci-dessous montre l'utilisation du sous-programme DB25. Dans la première image on voit les champs remplis pour chaque variable.




Cette image montre le parcours d'outil du sous-programme DB25.




Cette image montre l'action du bouton *Nouveau* et de l'onglet personnalisé pour créer très facilement la découpe de trois DB25 en un seul programme.




## 4.7 Création d'un sous-programme

- Pour la création d'un sous-programme à utiliser avec Ngcgui, le nom de fichier et le nom du sous-programme doivent être les mêmes.
- Le fichier doit être placé dans le sous-répertoire pointé dans le fichier ini.
- À la première ligne peut se trouver un commentaires de type info: qui doit être placé au début du sous-programme.
- Le sous-programme doit être entouré par les balises `sub` et `endsub`.
- Les variables utilisées doivent être des variables numérotées et ne doivent pas sauter de numéro.
- Des commentaires et presets peuvent être inclus.

```
(info: simp -- simple exemple de sous-programme -- Ctrl-U pour éditer)
o<simp> sub
  #<ra>       = #1 (=.6 Rayon A) ;Example de èparamtre avec un commentaire
  #<radius_b> = #2 (=0.4)         ;Example de èparamtre sans commentaire
  #<feedrate> = #3 (Feedrate)     ;Example de èparamtre sans preset
  g0x0y0z1
  g3 i#<ra> f#<feedrate>
  g3 i[0-#<radius_b>]
o<simp> endsub
```

# Chapitre 5

# L'interface graphique Touchy

Touchy est une interface utilisateur pour LinuxCNC, destinée à être utilisée avec les panneaux de commande de machines. Il ne nécessite ni clavier, ni souris.

Il a été conçu pour fonctionner également sur les écrans tactiles, en combinaison avec une manivelle électronique (MPG) et des boutons et interrupteurs.

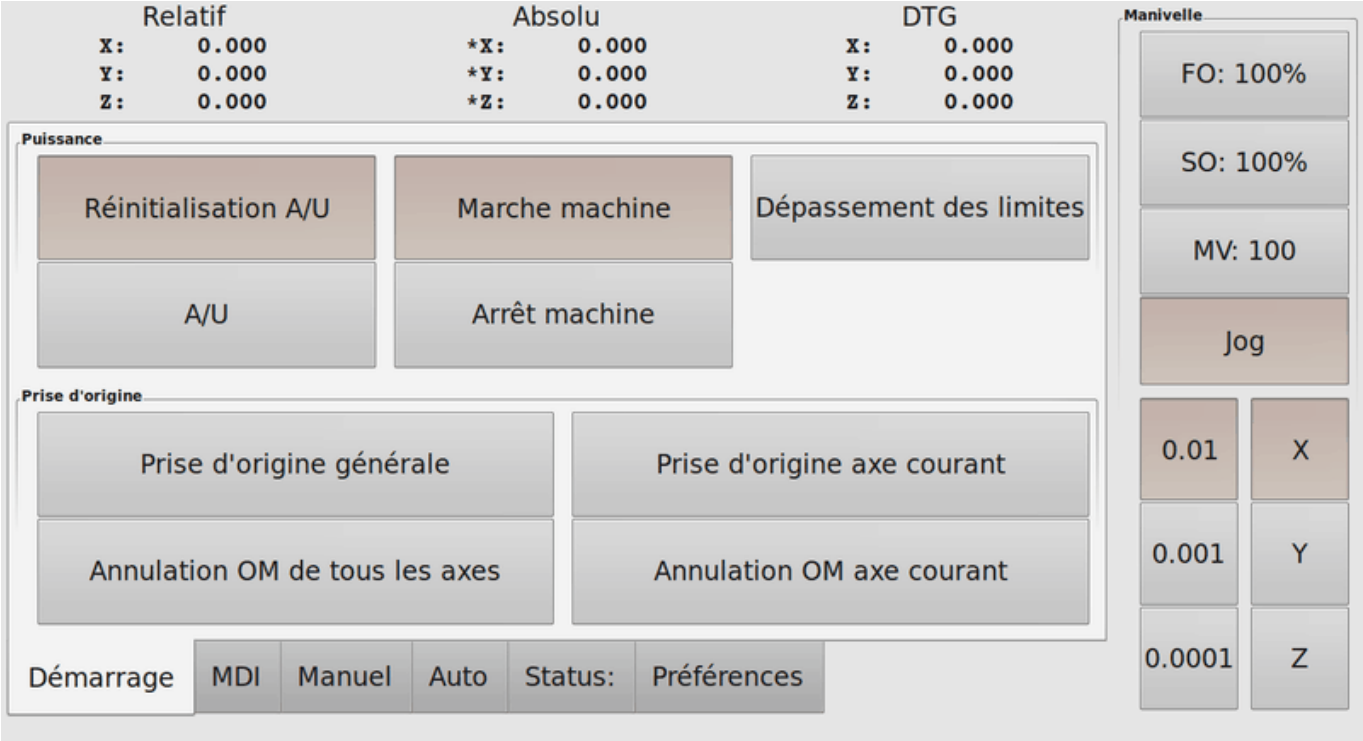



FIGURE 5.1 – L'interface tactile Touchy

## 5.1 Panneau de Configuration

### 5.1.1 Connections avec HAL

Touchy requiert qu'un fichier nommé *touchy.hal* soit créé, dans le même répertoire de configuration que le fichier ini, pour y connecter ses contrôles. Touchy exécute les commandes de HAL dans ce fichier après qu'il a créé ses propres pins, lesquelles

sont disponibles pour connexion.

Touchy dispose de plusieurs pins de sortie destinées à être connectées au contrôleur de mouvement pour gérer les manivelles de jog:
- touchy.jog.wheel.increment, qui doit être connecté à la pin *axis.N.jog-scale* de chacun des N axes.
- touchy.jog.wheel.N, qui doit être connecté à *axis.N.jog-enable* pour chacun des N axes.
- En plus d'être connecté à *touchy.wheel-counts*, le compteur d'impulsions de la manivelle doit aussi être connecté à *axis.N.jog-counts* pour chacun des N axes. Si le composant de HAL *ilowpass* est utilisé pour adoucir les mouvements de jog à la manivelle, il faut l'appliquer uniquement sur *axis.N.jog-counts* et non sur *touchy.wheel-counts*.

#### 5.1.1.1 Contrôles requis

- Un bouton *Abandon* (contact momentané) connecté sur la HAL pin *touchy.abort*
- Un bouton de *Départ cycle* (contact momentané) connecté à *touchy.cycle-start*
- Volant/Manivelle/MPG, connecté à *touchy.wheel-counts* et à la pin de mouvement comme décrit précédemment.
- Un bouton à bascule simple (contact à deux positions) connecté à *touchy.single-block*

#### 5.1.1.2 Contrôles optionnels

- Pour le jog continu, un interrupteur à trois positions avec retour au centre (ou deux boutons momentanés) pour chacun des axes concernés, attaché à *touchy.jog.continuous.x.negative* et à *touchy.jog.continuous.x.positive*, etc. pour les autres axes.
- Si un bouton de genouillère est nécessaire, (pour jogger Z en haut de sa course en grande vitesse), un bouton à contact momentané sera connecté à *touchy.quill-up*.

#### 5.1.1.3 Voyants de panneau optionnels

- *touchy.jog.active* indique quand les contrôles de jog du panneau sont actifs.
- *touchy.status-indicator* est allumé en continu quand la machine exécute un G-code et clignote quand la machine est en marche mais en pause, ou en vitesse à zéro.

### 5.1.2 Recommandé pour toutes les configurations

- Un bouton d'Arrêt d'Urgence (A/U) câblé physiquement, dans la chaîne d'arrêt d'urgence.

## 5.2 Réglages

Pour utiliser Touchy, dans la section [DISPLAY] du fichier ini de la machine, modifier la ligne de cette manière: DISPLAY = touchy

Quand Touchy démarre pour la première fois, vérifier l'onglet *Préférences*. Dans le cas d'un écran tactile, choisir de cacher le pointeur dans les options, pour obtenir les meilleurs résultats.

La fenêtre d'état est fixée en haut, ajustée par la taille d'une police fixe. La résolution de Gnome peut affecter cela. Si le bas de l'écran est coupé, aller dans le gestionnaire de résolution de Gnome pour revenir au réglage d'origine, à 96 DPI.

### 5.2.1 Macros

Touchy peut invoquer les macros avec mots O au travers de l'interface MDI. Pour configurer cette possibilité, dans la section [TOUCHY] du fichier ini, ajouter une ou plusieurs lignes avec MACRO. Chacune de ces lignes doit respecter le format suivant:

```
MACRO=increment xinc yinc
```

Dans lequel, *increment* est le nom de la macro, laquelle accepte deux paramètres, nommés ici *xinc* et *yinc*.

Maintenant, placer la macro proprement dite dans un fichier nommé *increment.ngc*. La variable PROGRAM_PREFIX du fichier ini pourra être utilisée pour identifier le répertoire contenant ce fichier. Il est également possible de le déclarer dans la variable SUBROUTINE_PATH.

Elle pourrait ressembler à cela:

```
O<increment> sub
G91 G0 X#1 Y#2
G90
O<increment> endsub
```

Noter que le nom du sous-programme, le nom de la macro ainsi que le nom du fichier .ngc doivent correspondre exactement, y compris les minuscules/ majuscules des noms.

Quand la macro est invoquée en pressant le bouton *Macro* dans l'onglet MDI de Touchy, il est possible de saisir des valeurs pour xinc et yinc, lesquelles seront passées à la macro comme étant respectivement **#1** et **#2**. Les paramètres laissées vides sont passés comme des valeur **0**.

Si il y a plusieurs macros différentes, presser le bouton *Macro* répétitivement pour les faire défiler.

Dans notre petit exemple, si -1 est entré pour xinc puis que le *Départ cycle* est pressé, un **G0**, sera invoqué, provoquant un déplacement en vitesse rapide, vers la gauche, de une unité machine.

Cette capacité d'utilisation des macros est très utile pour le palpage de contours ou d'orifices ainsi que pour d'autres opérations simples pré-configurées, de fraisage ou de perçage, qui pourront être réalisées depuis le panneau de Touchy sans avoir, pour cela, à écrire de programme G-code.

# Chapitre 6

# L'interface graphique TkLinuxCNC

## 6.1 Introduction

TkLinuxCNC est l'interface utilisateur graphique la plus populaire après Axis, c'est l'interface traditionnelle de LinuxCNC. Elle est écrite en Tcl et utilise le toolkit Tk pour l'affichage. Le fait d'être écrite en TCL la rend vraiment très portable (elle fonctionne sur une multitude de plateformes).

FIGURE 6.1 – L'affichage de TkLinuxCNC

## 6.2 Utiliser TkLinuxCNC

Pour sélectionner l'interface graphique TkLinuxCNC avec LinuxCNC, éditer le fichier .ini et dans la section [DISPLAY] modifier l'affichage comme ci-dessous:

```
DISPLAY = tklinuxcnc
```

Puis, lancer LinuxCNC et choisir ce fichier ini. La configuration qui se trouve dans *sim/tklinuxcnc/tklinuxcnc.ini* est déjà configurée pour utiliser TkLinuxCNC comme interface utilisateur.

Quand LinuxCNC est lancé avec TkLinuxCNC, une fenêtre comme celle-ci s'affiche.

### 6.2.1 Une session typique avec TkLinuxCNC

1. Lancer LinuxCNC et sélectionner un fichier de configuration.
2. Libérer l'*Arrêt d'Urgence* et mettre la machine en marche (en pressant F1 puis F2).
3. Faire l'*Origine Machine* de chacun des axes.
4. Charger un fichier d'usinage.
5. Brider le brut à usiner sur la table.

6. Faire l'*Origine Pièce* de chacun des axes, à l'aide du jog ou en introduisant une valeur de décalage d'origine après un clic droit sur le nom d'un axe.
7. Lancer le programme.
8. Pour refaire une autre pièce identique, reprendre à l'étape 6. Pour usiner une pièce différente, reprendre à l'étape 4. Quand c'est terminé, quitter LinuxCNC.

## 6.3 Éléments affichés par TkLinuxCNC

La fenêtre TkLinuxCNC contient les éléments suivants:
– Une barre de menu permettant diverses actions;
– Un jeu de boutons permettant d'agir sur le mode de travail, Marche/Arrêt de la broche et autres éléments;
– Une barre de statut pour l'affichage des différents offsets;
– Une zone d'affichage des coordonnées;
– Un jeu de curseurs pour contrôler la *vitesse de jog*, le *Correcteur de vitesse d'avance* et le *Correcteur de vitesse broche* qui permettent d'augmenter ou de diminuer ces vitesses ;
– Une boîte d'entrée de données manuelles;
– Une barre de statut affichant le bloc de programme actif, G-codes, M-codes, mots F et S;
– Les boutons relatifs à l'interpréteur;
– Une zone d'affichage de texte montrant le G-code du programme chargé.

### 6.3.1 Boutons principaux

Dans la première ligne de la gauche vers la droite et cycliquement:

1. Marche Machine: *Arrêt d'Urgence Arrêt d'Urgence relâché* / *Marche*
2. Bascule gouttelettes
3. Broche moins vite
4. Direction de rotation de la broche *Arrêt broche* / *Broche sens horaire* / *Broche sens anti-horaire*
5. Broche plus vite
6. Annuler

puis dans la deuxième ligne:

1. Mode de marche: *MANUEL* / *MDI* / *AUTO*
2. Bascule d'arrosage
3. Bascule du contrôle frein de broche

### 6.3.2 Barre de statut des différents offsets

Elle affiche, l'offset de rayon de l'outil courant (sélectionné avec Txx M6), l'offset éventuel de longueur d'outil si il est actif et les offsets de travail (ajustables par un clic droit sur les coordonnées).

### 6.3.3 Zone d'affichage des coordonnées

La partie principale affiche la position courante de l'outil. La couleur varie selon l'état de l'axe. Si l'axe n'est pas référencé il est affiché en caractères jaunes. Si il est référencé il s'affiche en vert. Si il est en erreur, TkLinuxCNC l'affiche en rouge pour montrer un défaut. (par exemple si un contact de fin de course est activé).

Pour interpréter correctement les différentes valeurs, se référer aux boutons de droite. Si la position est *Machine*, alors la valeur affichée est en coordonnées machine. Si elle est *Relative*, la valeur affichée est en coordonnées pièce. Deux autres en dessous indiquent *actuelle* ou *commandée*. Actuelle fait référence aux valeurs retournées par les codeurs (si la machine est équipée de servomoteurs) et *commandée* fait référence à la position à atteindre envoyée aux moteurs. Ces valeurs peuvent différer pour certaines raisons: Erreur de suivi, bande morte, résolution d'encodeur ou taille de pas. Par exemple, si un mouvement est commandé

vers X0.08 sur une fraiseuse, mais qu'un pas moteur fait 0.03, alors la position *Commandée* sera 0.03 mais la position *Actuelle* sera soit 0.06 (2 pas) soit 0.09 (3 pas).

Deux autres boutons permettent de choisir entre la vue *Articulation* et la vue *Globale*. Cela a peu de sens avec les machines de type normal (cinématiques triviales), mais se révèle très utile sur les machines avec des cinématiques non triviales telles que les robots ou plateforme de Stewart. (Des informations plus complètes se trouvent dans le manuel de l'intégrateur).

#### 6.3.3.1 Parcours d'outil

Quand la machine se déplace, elle laisse un tracé appelé parcours d'outil. La fenêtre d'affichage du parcours d'outil s'active via le menu *Vues* → *Parcours d'outil*.

### 6.3.4 Contrôle en automatique

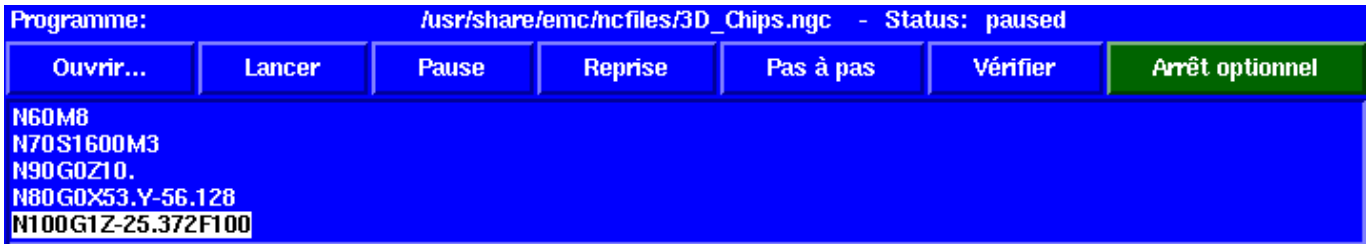



FIGURE 6.2 – Interpréteur de TkLinuxCNC

#### 6.3.4.1 Boutons de contrôle

Les boutons de contrôle de la partie inférieure de TkLinuxCNC, visibles sur l'image ci-dessus, sont utilisés pour l'exécution du programme:
- *Ouvrir* pour charger un fichier,
- *Lancer* pour commencer l'usinage,
- *Pause* pour stopper temporairement l'usinage,
- *Reprise* pour reprendre un programme mis en pause,
- *Pas à pas* pour avancer d'une seule ligne de programme,
- *Vérifier* pour vérifier si il contient des erreurs,
- *Arrêt optionnel* pour basculer l'arrêt optionnel, si ce bouton est vert l'exécution du programme est stoppée quand un code M1 est rencontré.

#### 6.3.4.2 Zone texte d'affichage du programme

Quand un programme est lancé, la ligne courante est affichée en surbrillance blanche. L'affichage du texte défile automatiquement pour montrer la ligne courante.

### 6.3.5 Contrôle en manuel

#### 6.3.5.1 Touches implicites

TkLinuxCNC permet les déplacements manuels de la machine. Cette action s'appelle le *jog*. Premièrement, sélectionner l'axe à déplacer en cliquant dessus. Puis, cliquer et maintenir les boutons **+** ou **-** selon la direction du mouvement souhaité. Les quatre premiers axes peuvent aussi être déplacés à l'aide des touches fléchées pour les axes X et Y, Pg.préc et Pg.suiv pour l'axe Z et les touches [ et ] pour l'axe A.

Si *Continu* est activé, le mouvement sera continu tant que la touche sera pressée, si une valeur d'incrément est sélectionnée, le mobile se déplacera exactement de cette valeur à chaque appui sur la touche ou à chaque clic. Les valeurs disponibles sont:

```
1.0000 0.1000 0.0100 0.0010 0.0001
```

En cliquant le bouton *Origine* ou en pressant la touche Origine, l'axe actif est référencé sur son origine machine. Selon la configuration, la valeur de l'axe peut être simplement mise à la position absolue 0.0, ou la machine peut se déplacer vers un point spécifique matérialisé par le *contact d'origine*. Voir le manuel de l'intégrateur pour plus de détails sur les prises d'origine.

En cliquant le bouton *Dépassement de limite*, la machine permet un jog temporaire pour même si l'axe à franchi une limite d'axe fixée dans le fichier .ini. Noter que si *Dépassement de limite* est activé il s'affiche en rouge.

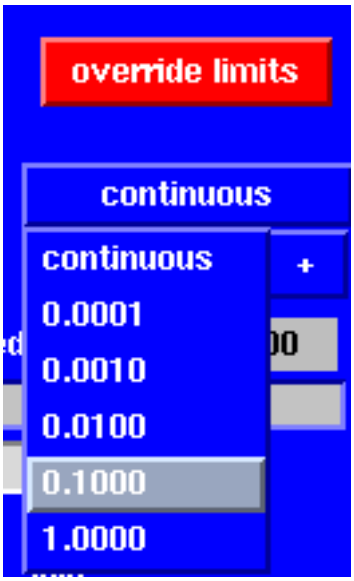



FIGURE 6.3 – Exemple de dépassement de limite et incréments de jog

#### 6.3.5.2 Le groupe de boutons *Broche*

Le bouton central du dessus sélectionne le sens de rotation de la broche: Anti-horaire, Arrêt, Horaire. Les boutons fléchés augmentent ou diminuent la vitesse de rotation. Le bouton central du dessous permet d'engager ou de relâcher le frein de broche. Selon la configuration de la machine, les items de ce groupe ne sont peut être pas tous visibles.

#### 6.3.5.3 Le groupe de boutons *Arrosage*

Ces deux boutons permettent d'activer ou non les lubrifiants *Gouttelettes* et *Arrosage*. Selon la configuration de la machine, les items de ce groupe ne sont peut être pas tous visibles.

### 6.3.6 Entrée manuelle de G-code (MDI)

L'entrée manuelle de données (aussi appelée MDI), permet d'entrer et d'exécuter des lignes de G-code, une à la fois. Quand la machine n'est pas en marche ni mise en mode MDI, l'entrée de code n'est pas possible.

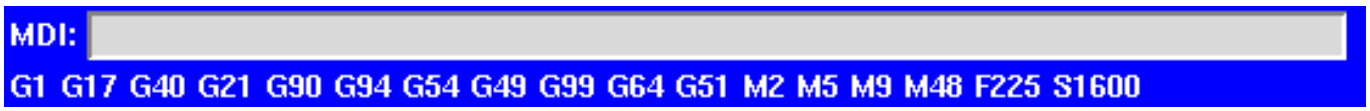



FIGURE 6.4 – Le champ de saisie des entrées manuelles

#### 6.3.6.1 MDI:

Le mode MDI permet d'exécuter une commande en G-code en pressant la touche *Entrée*.

#### 6.3.6.2 G-Codes actifs

Ce champs montre les *codes modaux* actuellement actifs dans l'interpréteur. Par exemple, **G54** indique que le système de coordonnées courant est celui de G54 et qu'il s'applique à toutes les coordonnées entrées.

### 6.3.7 Vitesse de Jog

En déplaçant ce curseur, la vitesse de jog peut être modifiée. Le nombre indique une vitesse en unités par minute. Le champs de texte est cliquable. Un clic ouvre un dialogue permettant d'entrer un nombre.

### 6.3.8 Correcteur de vitesse d'avance travail

En déplaçant ce curseur, la vitesse d'avance travail peut être modifiée. Par exemple, si la vitesse d'avance travail du programme est **F600** et que le curseur est placé sur 120%, alors la vitesse d'avance travail sera de 720. Le champs de texte est cliquable. Un clic ouvre un dialogue permettant d'entrer un nombre.

### 6.3.9 Correcteur de vitesse de broche

Le fonctionnement de ce curseur est le même que celui de la vitesse d'avance, mais il contrôle la vitesse de rotation de la broche. Si le programme demande S500 (broche à 500 tr/mn) et que le curseur est placé sur 80%, alors la vitesse de broche résultante sera de 400 tr/mn. Le minimum et le maximum pour ce curseur sont définis dans le fichier ini. Par défaut le curseur est placé sur 100%. Le champs de texte est cliquable. Un clic ouvre un dialogue permettant d'entrer un nombre.

## 6.4 Raccourcis clavier

La plupart des actions de TkLinuxCNC peuvent être accomplies au clavier. Beaucoup des raccourcis clavier ne sont pas accessibles en mode MDI.

Les raccourcis clavier les plus fréquemment utilisés sont montrés dans la table ci-dessous.

TABLE 6.1: Les raccourcis clavier les plus utilisés

| **Touche** | **Action** |
|---|---|
| F1 | Bascule de l'Arrêt d'Urgence |
| F2 | Marche/Arrêt machine |
| *, 1 .. 9, 0 | Correcteur vitesse d'avance 0% à 100% |
| X, * | Active le premier axe |
| Y, 1 | Active le deuxième axe |
| Z, 2 | Active le troisième axe |
| A, 3 | Active le quatrième axe |
| Origine | POM de l'axe actif |
| Gauche, Droite | Jog du premier axe |
| Haut, Bas | Jog du deuxième axe |
| Pg.prec, Pg.suiv | Jog du troisième axe |
| [, ] | Jog du quatrième axe |
| Echap | Arrête l'exécution |

# Chapitre 7

# L'interface graphique MINI

## 7.1 Introduction

[1]

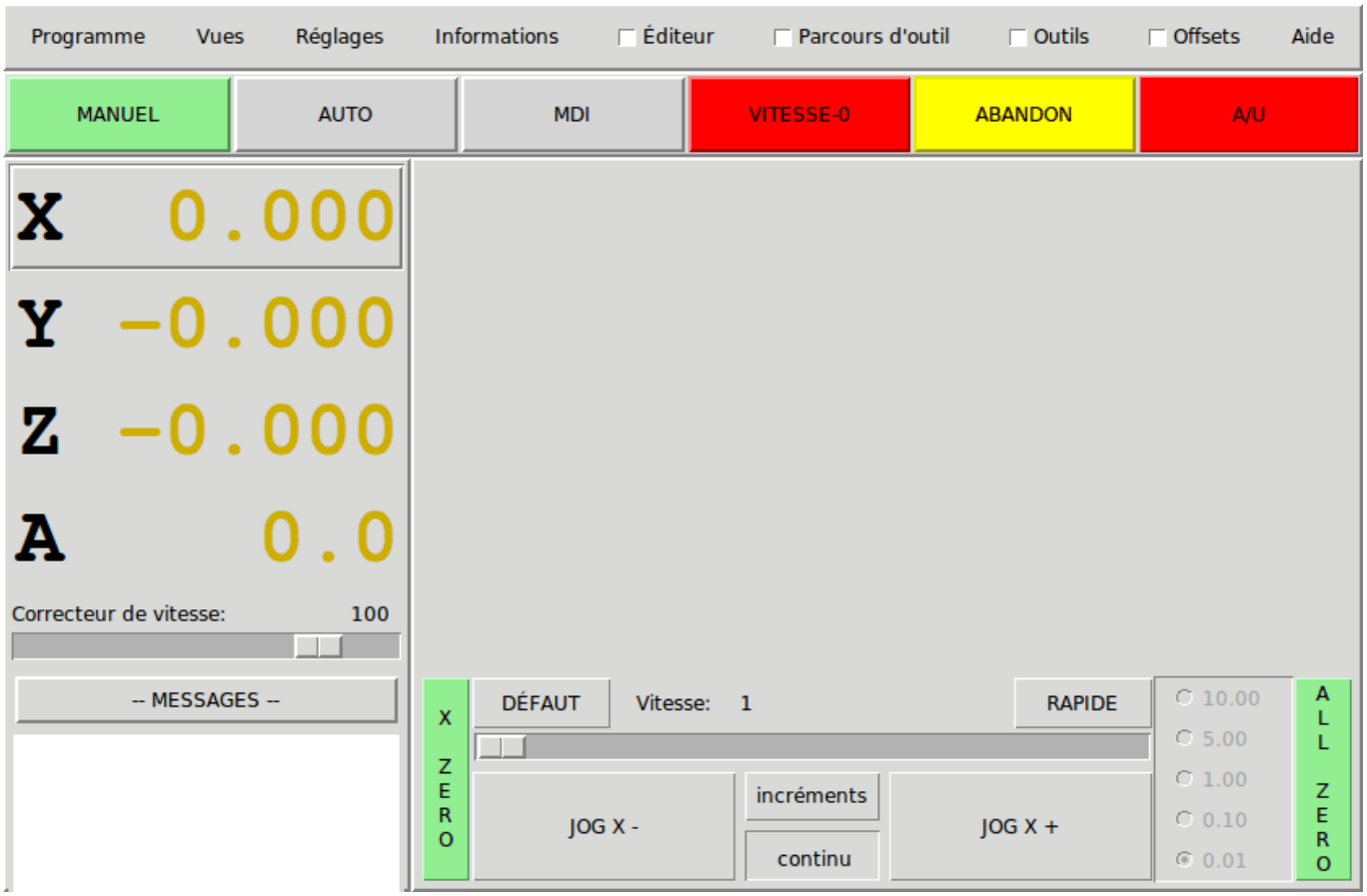



FIGURE 7.1 – L'interface graphique Mini

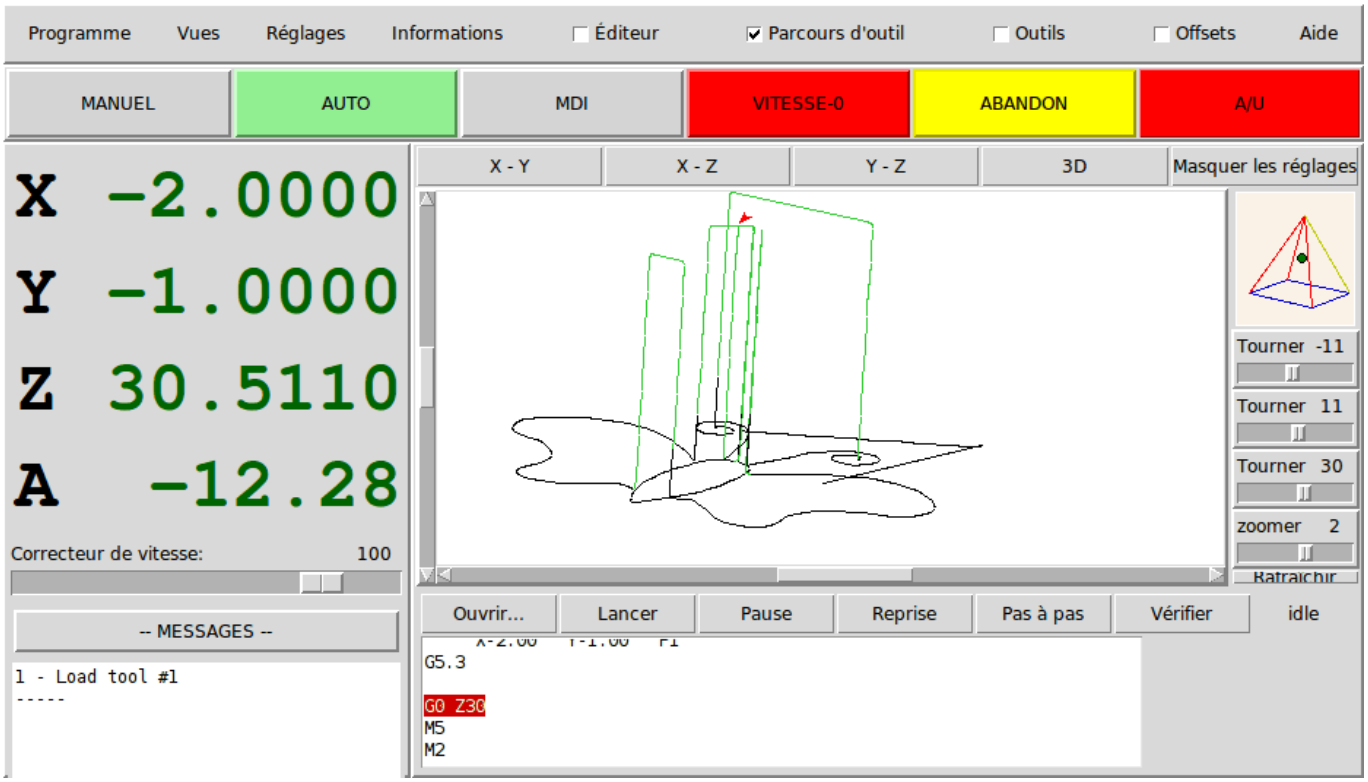
Mini a été prévu pour être une interface graphique plein écran. Il a été écrit initialement pour Sherline CNC mais est disponible pour ceux qui veulent l'utiliser, le copier le distribuer selon les termes de la GPL.

1. Une grande partie de ce chapitre provient du manuel Sherline CNC operators.

Au lieu d'ouvrir une nouvelle fenêtre pour chaque chose que l'opérateur veux faire, Mini permet de les afficher toutes dans la même fenêtre. Certaines parties de ce chapitre sont des copies des instructions écrites par Joe Martin et Ray Henry.

## 7.2 L'écran

**Affichage de Mini lors du fonctionnement de LinuxCNC**




L'écran de Mini est visible dans plusieurs sections, voir la figure ci-dessus. Il inclu une barre de menu, un jeu de boutons de contrôle juste en dessous et deux larges colonnes d'information montrant l'état de la machine et permettant à l'opérateur d'entrer des commandes ou des programmes.

Quand on compare entre l'écran d'accueil et l'écran en cours de fonctionnement, des différences apparaissent. Sur la seconde image:
- Chaque axe a été référencé sur son origine — L'affichage de la visu est vert sombre
- LinuxCNC est en mode auto — le bouton est vert clair
- Le parcours d'outil a été activé — le parcours est visible dans la fenêtre
- Le bloc de G-code en cours est en surbrillance blanche sur fond rouge parmi les lignes défilantes du programme.

Dès le début de l'utilisation de Mini on découvre combien il est facile de visualiser le fonctionnement de LinuxCNC et d'agir sur lui.

## 7.3 Barre de menu

La première ligne est occupée par la barre de menu. Elle permet de configurer les informations optionnelles que l'écran doit afficher. Certains des items du menu sont très différents de ce qu'il est coutume de rencontrer avec d'autres programmes. Il faut prendre quelques minutes pour regarder chaque item du menu et se familiariser avec les possibilités qu'il offre.

Le menu inclus chacune des sections et sous-sections suivantes:

**Programme**

ce menu inclus les deux fonctions *reset* et *quitter*. Reset renvoie LinuxCNC dans l'état dans lequel il était quand il à démarré. Certains items de conditions de démarrage comme les unités du programme peuvent être spécifiés dans le fichier ini.

**Vues**

ce menu inclus plusieurs éléments d'affichage pouvant être ajoutés pendant qu'un programme s'exécute, pour donner des information supplémentaires. Ce qui inclus:

**Type_de_position**
cet item ajoute une ligne, au dessus de l'afficheur principal de position, qui indique si la position affichée est en coordonnées machine (absolues) ou en coordonnées pièce (relatives), si la position affichée est actuelle ou commandée et si l'unité et le pouce le millimètre. Ce qui peut être modifié en agissant sur le menu *Réglages* décrit plus loin.

**Informations_outil**
cet item ajoute une ligne immédiatement sous l'afficheur principal, elle affiche quel outil est selectionné et la valeur de l'offset appliqué.

**Informations_offset**
ajoute une ligne immédiatement sous celle des informations outil, elle affiche les offsets de travail actuellement appliqués. C'est la distance totale pour chaque axe depuis son point d'origine machine.

**Afficher_Reprise**
ajoute, en mode Auto, un bloc de boutons en bas à droite de l'affichage du programme. Ces boutons permettent à l'opérateur de relancer un programme après un A/U ou un Abandon. Le programme est figé lorsqu'un A/U ou un Abandon est engagé, mais l'opérateur peut ici choisir le bloc sur lequel redémarrer dans le mode Auto, en activant cet item de menu.

**Masquer_Reprise**
enlève le bloc de boutons contrôlant la reprise d'un programme stoppé par Abandon ou Arrêt d'Urgence.

**Diviser_la_vue_droite**
modifie la nature de l'affichage de la colonne de droite pour qu'il montre plus d'informations.

**Afficher_en_mode_complet**
modifie l'affichage de la colonne de droite, soit en mode boutons et la colonne divisée, soit affichage sur la colonne entière. En mode manuel et en utilisant ce mode entier, les boutons de la broche et du lubrifiant sont visibles en plus des boutons de mouvement.

**Afficher_dans_toute_la_fenêtre**
modifie l'affichage dans la colonne de droite pour qu'elle soit remplie entièrement.

**Réglages**

Les items de ce menu permettent à l'opérateur de contrôler certains paramètres en cours d'usinage.

**Position_Actuelle**
régle l'afficheur principal sur des valeurs de position actuelle (basées sur la machine).

**Position_Commandée**
régle l'afficheur principal sur des valeurs de position commandée (positions cibles).

**Position_Machine**
régle l'afficheur principal sur des valeurs de position absolue, distance depuis le point d'origine machine.

**Position_Relative**
régle l'affichage principal pour qu'il affiche les valeurs de la position courante en incluant tous les offsets comme les origines pièces qui seraient actives. Pour plus de détails voir le chapitre sur les *Systèmes de coordonnées*.

**Informations**

Indique diverses activités en affichant des valeurs dans la zône MESSAGES.

**Fichier_programme**
Indique le nom du fichier programme actif.

**Editeur_de_Fichiers**
Indique le nom de l'éditeur de texte lancé quand un fichier est choisi pour être édité.

**Fichier_de_paramètres**
Indique le nom du fichier devant être utilisé pour enregistrer les paramètres du programme. Il est possible d'avoir plus d'informations dans le chapitre sur les offsets et l'utilisation des paramètres dans les programmes.

**Fichiers_d'outils**
Indique le nom du fichier d'outils utilisé pour l'usinage en cours.

**G-Codes_actifs**
Indique la liste des G-codes modaux actifs au moment ou cet item a été sélectionné. Pour plus d'informations sur les codes modaux, se reporter aux chapitres sur la programmation.

**Aide**
Ouvre une fenêtre de texte contenant un fichier d'aide.

Noter que entre le menu informations et le menu d'aide il y a un jeu de quatre cases à cocher. Elles sont appelées cases sélectionnables car elles passent au rouge si elles sont sélectionnées. Ces quatre cases, Editeur, Parcours d'outil, Outils et Offsets activent les différents écrans. Si plus d'une de ces cases est validée (case en rouge) il est possible de passer de l'un à l'autre des écrans surgissant par un clic droit de la souris.

## 7.4 Barre de boutons de contrôle

Sous la ligne de menu se trouve une ligne de boutons de contrôle. Ce sont les boutons principaux de l'interface. En utilisant ces boutons il est possible de changer de mode cycliquement entre *MANUEL* puis *AUTO* puis *MDI* (Manual Data Input). Ces boutons ont un fond vert clair quand le mode correspondant est actif.

Il est également possible d'utiliser les boutons *VITESSE-0*, *ABANDON* et *A/U* pour contrôler les mouvements du programme.

### 7.4.1 MANUEL

Ce bouton, ou la touche **F3** placent LinuxCNC en mode Manuel et affiche un jeu condensé de boutons que l'opérateur peut utiliser pour effectuer des commandes de mouvements manuels. Les labels des boutons de jog varient pour refléter l'axe actif. Quand en plus de ce mode manuel, la vue en mode complet est activée, les boutons de la broche et du lubrifiant sont visibles en plus de ceux de mouvement. La touche **i** ou **I** basculera entre jog en continu et jog par incréments. Presser cette touche plusieurs fois provoquera le changement cyclique de la taille de l'incrément de jog.

FIGURE 7.2 – Boutons du mode Manuel

Un bouton à été ajouté pour désigner la position actuelle comme étant la position d'origine. Une machine simple telle que la Sherline 5400 serait facile à utiliser sans avoir de position d'origine. Ce bouton met les offsets de tous les axes à zéro et place l'origine de tous les axes au point courant.

Ce concentrer sur les axes est important ici. Noter que sur la capture du début de chapitre qui est en mode manuel, on voit un cadre autour de l'axe X pour mettre son affichage en évidence. Ce cadre (le focus) indique que l'axe X est l'axe actif. Il est l'axe cible pour les mouvements de jog faits par appui sur les boutons *plus* ou *moins*. Il est possible de changer l'axe actif en cliquant sur l'affichage d'un autre axe. C'est également possible en mode manuel en pressant la touche de son nom sur le clavier. La casse n'a pas d'importance **Y** ou **y** donneront le focus à l'axe Y. **A** ou **a** le donneront à l'axe A. Pour aider à se rappeller quel est l'axe actif avant de faire un jog, son nom est indiqué sur les boutons de jog.

LinuxCNC peut faire un jog (mouvement d'un axe particulier) aussi longtemps que le bouton est maintenu pressé, si le jog est réglé sur *continu* , ou il peut se déplacer d'une valeur prédéfinie, quand il est réglé sur *incrément* s. Il est aussi possible de faire un jog de l'axe actif en pressant les touches **+** ou moins **-** du clavier. De nouveau, la casse n'a pas d'importance pour les jogs au clavier. Les deux petits boutons situés entre les deux gros du jog, permettent de choisir quel type de jog est souhaîté. Quand on est en mode incréments, les boutons de taille d'incréments sont accessibles. La sélection d'une taille d'incrément se fait par clic sur la case à cocher avec la souris ou cycliquement en pressant la touche **i** ou **I** sur le clavier. Le jog par incréments présente quelques effets intéressants autant qu'inattendus. Si vous pressez le bouton de jog alors qu'un mouvement de jog est déjà en cours, la

distance à laquelle il était lorsqu'est arrivée la commande du second jog, sera ajoutée à la position. Deux pressions successives rapides de 10mm d'incrément ne donneront pas 20mm de mouvement. Vous devez attendre que le premier soit terminé avant d'envoyer le second.

La vitesse de jog est affichée au dessus du curseur. Il est possible de régler le curseur en cliquant dans la glissière du curseur, du côté où vous voulez son déplacement, ou en cliquant sur les boutons *Défaut* ou *Rapide*. Ces réglages n'affectent que les mouvements de jog en mode manuel. Tant qu'un mouvement de jog est en cours, un changement de la vitesse de jog est sans effet sur le jog. Par exemple, disons que vous avez réglé le jog par incréments de 10mm. Même si vous pressez le bouton *Jog*, le déplacement de dix millimètres se finira à la vitesse à laquelle il a commencé.

### 7.4.2 AUTO

Quand le bouton Auto est pressé, ou **F4** sur le clavier, LinuxCNC passe dans ce mode, un jeu de boutons traditionnels des opérations en auto est affiché et une petite fenêtre textuelle s'ouvre, montrant une partie du programme. Durant un usinage, la ligne active est affichée en surbrillance blanche sur fond rouge.

Dans le mode auto, beaucoup de touches sont liées aux contrôles. Par exemple, les touches numérotées au dessus du clavier sont liées aux réglages du correcteur de vitesse d'avance travail. Le **0** l'ajuste à 100%, le **9** l'ajuste à 90% etc. D'autres touches fonctionnent de la même manière qu'avec l'interface TkLinuxCNC.

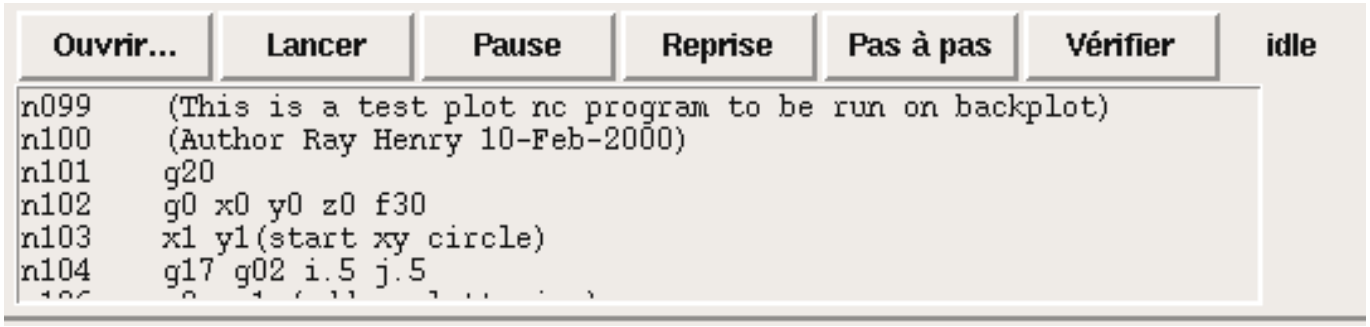



FIGURE 7.3 – Mode Auto

Le mode Auto n'affiche normalement pas les G-codes actifs ou modaux. Si l'opérateur veut les voir, il doit utiliser le menu *Informations → G-codes actifs* et afficher ainsi la liste des codes modaux dans la zône de texte MESSAGES.

Si un Arrêt d'Urgence ou un Abandon est pressé pendant un usinage, un jeu de boutons s'affiche en bas et à droite, ils permettent à l'opérateur de décaler la ligne de reprise vers l'avant ou vers l'arrière. Si la ligne de reprise n'est pas la dernière ligne active, elle sera mise en surbrillance blanche sur fond bleu. ATTENTION, une vitesse très faible et un doigt collé sur le bouton de pause est prudent pendant toute reprise de programme!

Ce qui est le mieux avec une machine CNC, c'est le mode auto. Le mode auto affiche les fonctions typiques que tout le monde espère utiliser avec LinuxCNC. Au dessus de la fenêtre un jeu de boutons qui contrôlent ce qui se passe en mode auto. En dessous, la fenêtre montrant la partie du programme en cours d'exécution. Quand le programme est lancé, la ligne active s'affiche en surbrillance blanche sur fond rouge. Les trois premiers boutons, *Ouvrir*, *Lancer* et *Pause* font ce que leurs noms indiquent. *Pause* stoppe le programme en cours d'exécution là où il est. Le bouton suivant, *Reprise*, reprend le mouvement. Le résultat est le même si le bouton *Vitesse-0* était pressé au lieu du bouton *Pause*, le mouvement est stoppé, *Pas à pas* reprend aussi le mouvement, mais il ne continue que jusqu'à la fin du bloc courant. Presser une nouvelle fois *Pas à pas* exécutera le mouvement du bloc suivant. Presser *Reprise* à ce moment là et l'interpréteur reviendra en arrière lire et relancer le programme. La combinaison entre *Pause* et *Pas à pas* marche un peu comme un seul bloc dans plusieurs interpréteurs. Avec la différence que *Pause* ne laisse pas le mouvement se poursuivre jusqu'à la fin du bloc courant. Le correcteur de vitesse d'avance travail peut se révéler très pratique pour s'approchez de la matière pour un premièr usinage. Le placer sur 100% pour les déplacements rapides, le régler sur 10% et basculez entre *Vitesse-0* et 10% en utilisant le bouton *Pause*. Quand vous êtes satisfait et que les choses se présentent bien, pressez le zéro à la droite du neuf en haut du clavier, et c'est parti.

Le bouton *Vérifier* passe le code dans l'interpréteur sans production de mouvements. Si Vérifier trouve un problème il stoppe la lecture juste derrière le bloc posant problème et affiche un message. La plupart du temps vous serez en mesure de régler le

problème avec votre programme par la lecture du message et en vérifiant la ligne de code en surbrillance dans la fenêtre du programme. Certains messages, toutefois, ne sont pas d'un grand secours. Parfois vous devrez lire une ou deux lignes en avant de celle en surbrillance pour voir le problème. Occasionnellement le message fait référence à quelque chose très en avant de la ligne en surbrillance. Le plus souvent ça se produit si vous oubliez de terminer votre programme par un code correct comme %, M2, M30, ou M60.

### 7.4.3 MDI

Le bouton MDI ou la touche **F5** activent le mode d'entrée de données en manuel (Manual Data Input). Ce mode affiche un simple champ de saisie d'une ligne de texte et montre les codes modaux actuellement actifs dans l'interpréteur.

Le mode MDI vous permet d'entrer de simples blocs et de les faire exécuter par l'interpréteur comme si ils étaient une partie d'un programme (Une sorte de programme d'une seule ligne). Vous pouvez exécuter des cercles, des arcs, des lignes et autres. Vous pouvez aussi mettre au point une ligne de programme en entrant cette ligne comme un seul bloc, attendre que le mouvement se termine et entrer le bloc suivant. Sous la fenêtre d'entrée, se trouve une liste des codes modaux courants. Cette liste peut être très pratique. J'oublie parfois d'entrer un G0 avant une commande de mouvement. Si rien ne se passe je regarde dans la liste si G80 est actif. G80 stoppe tous les mouvements. Si il y est je me rappelle de mettre un bloc comme G00 X0 Y0 Z0. Dans le MDI, vous entrez du texte depuis le clavier, ainsi toutes les touches principales ne fonctionnent pas comme raccourcis clavier pour les commandes machine. **F1** engage l'Arrêt d'Urgence.

Puisque les touches du clavier sont nécessaires à la saisie du texte, beaucoup des raccourcis clavier disponibles en mode auto ne le sont pas ici.

### 7.4.4 VITESSE-0 et CONTINUER

Vitesse-0 est une bascule. Quand LinuxCNC est prêt pour exécuter, ou qu'il exécute une commande de mouvement, ce bouton affiche son label *VITESSE-0* sur fond rouge. Si Vitesse-0 a été pressé il affiche le label *CONTINUER*. Utiliser ce bouton pour faire une pause dans un mouvement présente l'avantage d'être capable de relancer le programme d'où il a été stoppé. Vitesse-0 bascule simplement entre vitesse zéro et la vitesse d'avance travail avec l'éventuel correcteur qui était actif au moment de l'arrêt. Ce bouton et la fonction qu'il active sont également liés à la touche pause de la plupart des claviers.

### 7.4.5 ABANDON

Le bouton Abandon stoppe tous les mouvements quand il est pressé. Il désactive aussi la commande de marche de LinuxCNC. Plus aucun mouvement ne survient après l'appui sur ce bouton. Si vous êtes en mode auto, ce bouton enlève le reste du programme du planificateur de mouvements. Il enregistre aussi le numéro de la ligne qui s'exécutait quand il a été pressé. Vous pouvez vous servir de ce numéro de ligne pour redémarrer le programme après avoir supprimé la raison de l'appui…

### 7.4.6 Arrêt d'Urgence

Le bouton d'Arrêt d'Urgence est une bascule mais il a trois fonctionnements possibles.

- Au démarrage de Mini c'est un bouton avec le texte *A/U* écrit en noir sur fond rouge. C'est état de la machine est correct pour charger un programme ou faire un jog sur un axe. L'Arrêt d'Urgence est libéré quand il s'affiche dans cet état.
- Si vous pressez sur le bouton d'Arrêt d'Urgence pendant qu'un mouvement est exécuté, le texte sur le bouton devient *A/U Engagé* sur fond gris et le bouton s'enfonce. Plus aucun mouvement n'est possible et plus rien ne réagi sur l'interface Mini tant que l'Arrêt d'Urgence est dans cet état. Le presser à nouveau à la souris le fera repasser en conditions normales de fonctionnement.
- Un troisième état est encore possible. Un bouton enfoncé portant le texte *A/U Libéré* sur fond vert signifie que l'A/U a bien été libéré mais que la machine n'a pas été mise en marche. Normalement cet état apparaît quand l'A/U était libéré mais que la touche Marche Machine **F2** a été pressée.

Joe Martin disait, *Quand tout le reste a échoué, pressez un Arrêt d'Urgence software*. Si vous avez un circuit externe qui gère l'A/U en surveillant une broche du port parallèle ou celle d'une carte d'entrées/sorties, un arrêt d'urgence software pourra couper la puissance sur les moteurs.

La plupart du temps, quand un Abandon ou un Arrêt d'Urgence est engagé c'est parce que quelque chose se passe mal. Peut être un outil cassé et qui doit être changé. On passe alors en mode manual et on arrête la broche, on change l'outil et en supposant

que sa longueur reste la même, on est prêt pour relancer le programme. Si on renvoie l'outil à la même place ou il était avant, LinuxCNC va fonctionner parfaitement. Il est aussi possible de se déplacer sur la ligne suivante ou précédente de celle ou c'est produit l'abandon. Si vous pressez le bouton *Arrière* ou *Avant* vous voyez une ligne en surbrillance bleue montrant l'écart entre la ligne sur laquelle l'abandon s'est produit (restée en surbrillance rouge) et la ligne à laquelle LinuxCNC va redémarrer. En réfléchissant à ce qui va se produire au moment de la reprise vous serez en mesure de placer l'arête de l'outil là où la reprise pourra se faire de manière acceptable. Vous aurez peut être à solutionner certaines difficultés comme celles créées par les compensations de rayon d'outil le long d'une ligne diagonale et vous devrez être sûr de vous avant de presser sur le bouton *Reprise*.

## 7.5 Colonne de gauche

Il y a deux colonnes sous la barre de contrôle. La colonne de gauche affiche les informations intéressant l'opérateur. Il y a seulement deux boutons dans cette zône.

### 7.5.1 Afficheurs de position des axes

Ces afficheurs se comportent exactement comme ceux de TkeLinuxCNC. La couleur des afficheurs est importante.
– Rouge, elle indique que la machine est en appui sur un contact de fin de course ou que la polarité d'une limite est mal positionnée dans le fichier ini.
– Jaune, elle indique que la machine est prête pour faire ses prises d'origine.
– Verte, elle indique que la machine a bien été référencée sur ses points d'origine.
Le type de position affichée peut varier, selon les options choisies dans le menus *Réglages*. Les réglages par défaut, ou de démarrage, peuvent être changés dans le fichier ini pour correspondre à vos besoins.

### 7.5.2 Correcteur de vitesse travail

Immédiatement sous les afficheurs de position on trouve un curseur, c'est le correcteur de vitesse travail. Vous pouvez agir sur le correcteur de vitesse et sur le bouton Vitesse-0 dans tous les modes de marche. Le correcteur agit sur la vitesse de jog et sur la vitesse d'avance travail dans les modes manuel ou MDI. Il est possible de modifier la position du curseur en le déplaçant à la souris le long de sa glissière. Il est également possible de modifier le correcteur de 1% à chaque fois qu'un clic de souris est fait dans la glissière du curseur. En mode Manuel il est possible d'ajuster le correcteur par incréments de 10% avec les touches chiffrées du haut du clavier. Le curseur est une référence visuelle très pratique pour estimer la correction appliquée sur la vitesse d'avance programmée.

### 7.5.3 Messages

L'affichage des messages situé sous le curseur du correcteur de vitesse est une sorte de bloc-notes pour LinuxCNC. Si un problème survient, il est reporté sur ce bloc-notes. Si vous essayez de déplacer un axe alors que l'Arrêt d'Urgence est engagé, vous recevez un message qui dit quelques choses à propos des conditions de marche empêchant LinuxCNC de répondre à la commande de mouvement. Si un axe est en défaut, par exemple un dépassement de limite, le message affiché sur le bloc-notes indiquera ce qui se passe. Pour demander à l'opérateur de changer d'outil par exemple, vous pouvez aussi ajouter une ligne de code dans le programme qui s'affichera sur l'écran, dans la boîte de messages. Un exemple pourrait être: (msg, Montez l'outil N°3 puis pressez Reprise). Cette ligne de code, incluse dans un programme, va afficher *Montez l'outil 3 puis pressez Reprise* dans la boîte de messages. Le mot msg, (avec la virgule) est la commande qui fait apparaître le commentaire, sans *msg*, le message ne serait pas affiché.

Pour effacer les messages cliquer simplement sur le bandeau --MESSAGES-- au dessus du bloc-notes ou au clavier, presser **m** tout en maintenant la touche *Alt* appuyée.

## 7.6 Colonne de droite

La colonne de droite est l'emplacement sur lequel seront affichés les différents éléments résultants des choix de l'utilisateur. C'est ici que seront visibles les différentes entrées de texte, les différents affichages et les boutons du mode manuel. C'est ici

que le parcours d'outil sera affiché pendant l'exécution d'un programme. C'est également ici que l'éditeur de texte s'ouvrira pour permettre l'édition des programmes, l'édition des tables d'outils ou d'offsets. L'écran du mode manuel a déjà été décrit précédemment. Chaque écran surgissant va être décrit en détail ci-dessous.

### 7.6.1 Editeur de texte

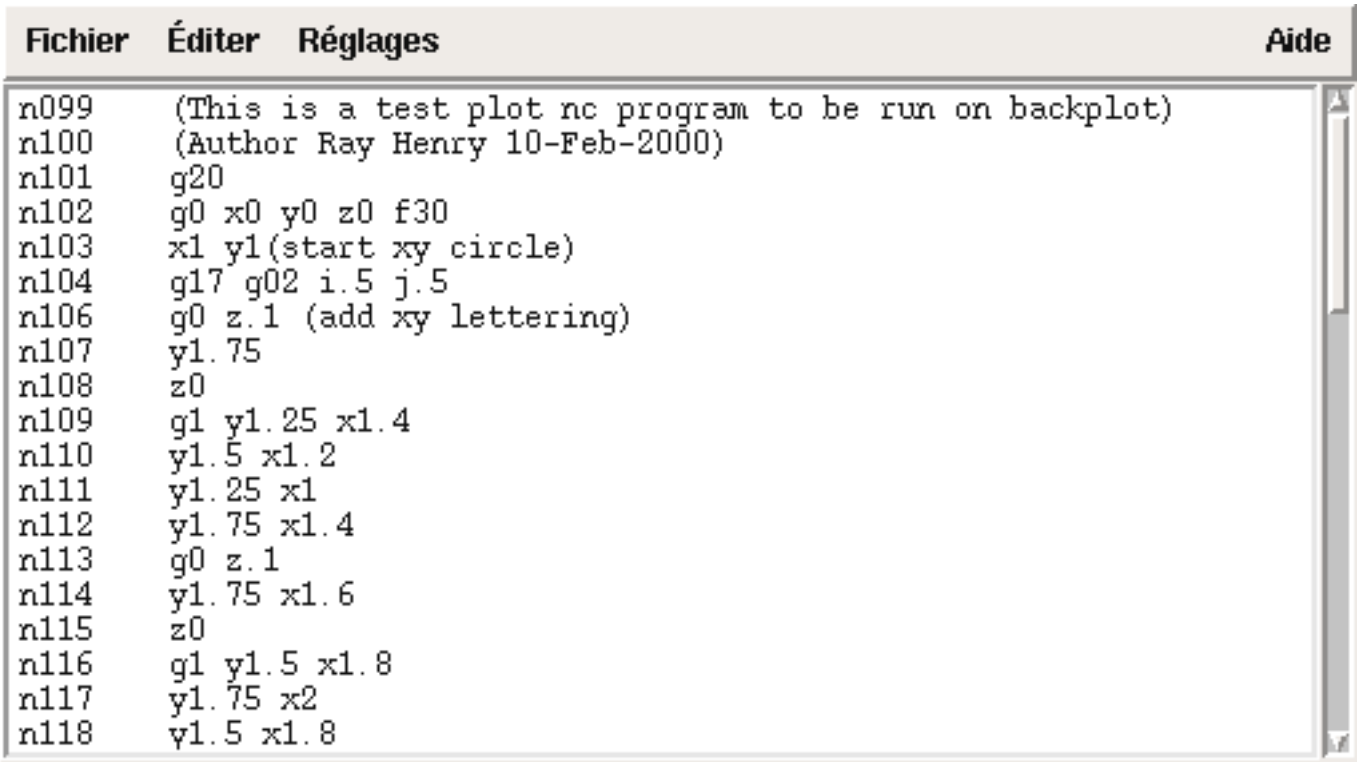



FIGURE 7.4 – Editeur de texte de Mini

L'éditeur de texte de Mini peut sembler un peu limité par rapport aux éditeurs de texte modernes. Il a été inclus parce qu'il comporte l'excellente possibilité de numéroter et renuméroter un programme de la même manière que l'interpréteur le fait avec un fichier. Il permet également le couper/coller d'une partie vers une autre du fichier. En plus, il permet d'enregistrer les changements faits dans le programme et de soumettre celui-ci à l'interpréteur de LinuxCNC depuis le même menu. Il est possible de travailler sur un fichier ouvert dans cet éditeur puis de l'enregistrer et de le recharger si LinuxCNC est en mode Auto. Si vous avez lancé un fichier et que vous avez besoin de l'éditer, ce fichier sera placé dans l'éditeur quand vous cliquez sur le bouton *Editeur* du menu.

### 7.6.2 Parcours d'outil

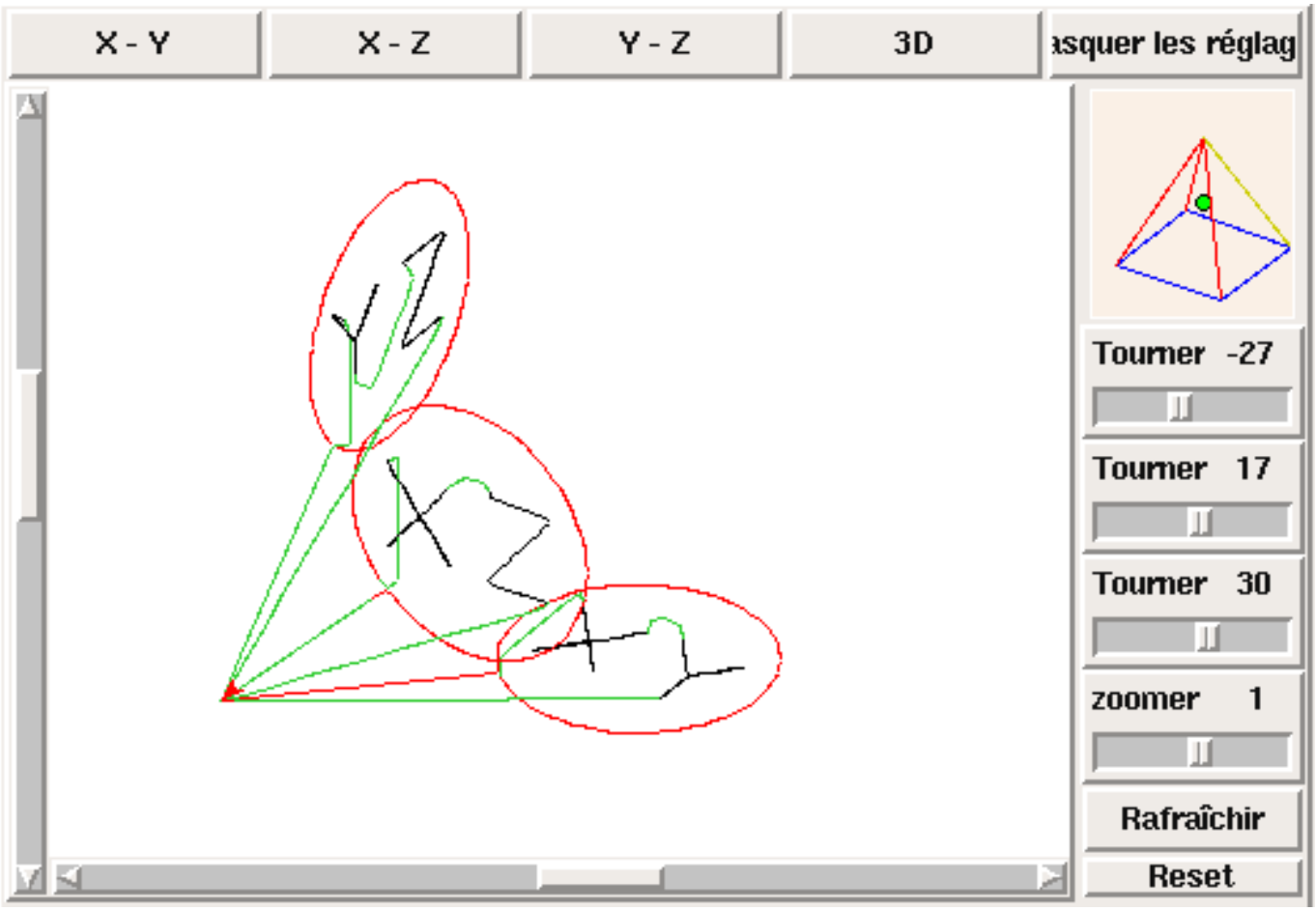



FIGURE 7.5 – Parcours d'outil de Mini

Le bouton *Parcours d'outil* du menu, affiche un tracé produit par le cheminement de l'outil pendant l'usinage. Cette trace est affichable selon plusieurs plans et en **3-D** qui est le mode par défaut. Les choix sont affichés au dessus et à droite de cette fenêtre. Si l'usinage est déjà en cours depuis un certain temps quand cette fenêtre est activée, il lui faudra un peu de temps pour recalculer la vue du parcours.

Sur le côté droit de la fenêtre une petite pyramide graphique montre l'angle depuis lequel l'opérateur voit le tracé du parcours d'outil. En dessous, une série de curseurs permettent de modifier les angles et la taille de la vue. Il est possible de contrôler les modifications en regardant l'attitude de la petite pyramide. Les modifications ne prendront effet qu'après appui sur le bouton *Rafraîchir*. Le bouton *Reset* efface toutes les traces du parcours affiché et prépare la fenêtre pour un nouveau lancement du programme, il conserve toutefois les orientations et le niveau de zoom actuellement définis.

Si un parcours d'outil est lancé avant qu'un programme ne soit démarré, il va afficher quelques lignes de couleur pour indiquer le type de tracé qui sera utilisé. Une ligne verte représente un mouvement en vitesse d'avance rapide. Une ligne noire, un mouvement en vitesse d'avance travail. Une ligne bleue et une ligne rouge indiquent respectivement un arc en sens anti-horaire et un arc en sens horaire.

Le tracé du parcours d'outil de Mini permet de zoomer et d'orienter les vues du programme en cours d'exécution mais il n'est pas conçu pour enregistrer un chemin d'outil sur une longue période de temps.

### 7.6.3 Page des outils

La page des outils permet d'ajuster la longueur et le diamètre des outils, ces valeurs deviennent effectives dès l'appui sur la touche *Entrée*. Il est nécessaire de définir les paramètres des outils avant de lancer le programme. Les offsets d'outils ne peuvent pas être modifiés une fois le programme démarré ni quand il est en pause.

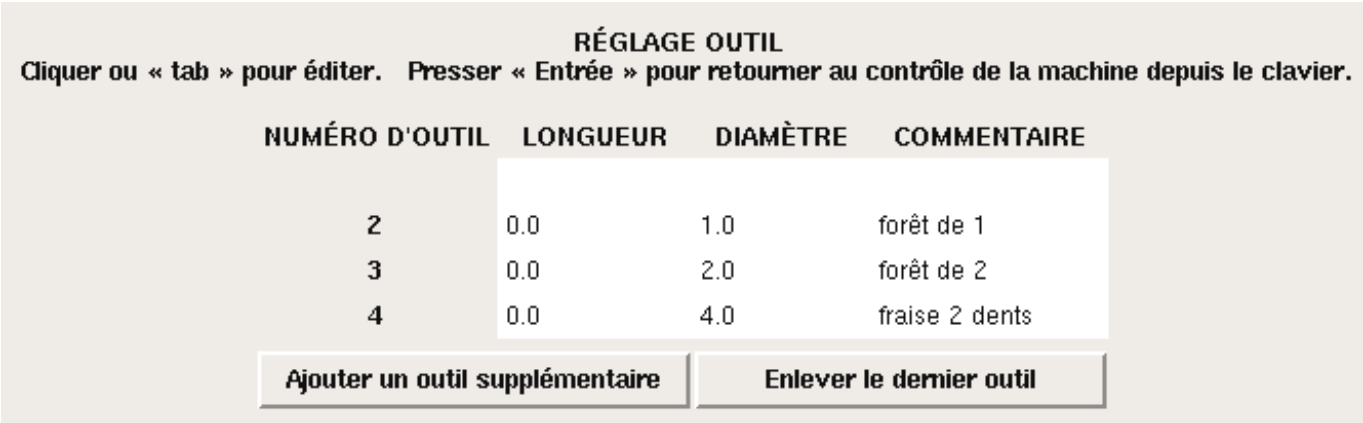



FIGURE 7.6 – Affichage de la table d'outils de Mini

Les boutons *Ajouter un outil* et *Enlever le dernier outil* portent sur la fin de la liste des outils de sorte que l'ajout d'outils s'effectuent vers le bas. Quand un nouvel outil est ajouté, il est utilisable dans le programme avec la commande G-code habituelle. Le nombre maximum d'outils est de 32 dans le fichier de configuration de LinuxCNC mais vous serez arrivé au bout des possibilités d'affichage de Mini bien avant celà. A la place vous pouvez utiliser le *menu* → *Vues* → *Afficher dans toute la fenêtre* pour voir plus d'outils si nécessaire.

### 7.6.4 Page des offsets

La page des offsets peut être utilisée pour afficher et ajuster les décalages d'origine des différents systèmes de coordonnées. Le système de coordonnées est choisi dans la colonne de gauche, par sélection d'une case à cocher. Quand un système de coordonnées est choisi il est possible d'entrer directement les valeurs ou de déplacer l'axe à une position d'apprentissage.

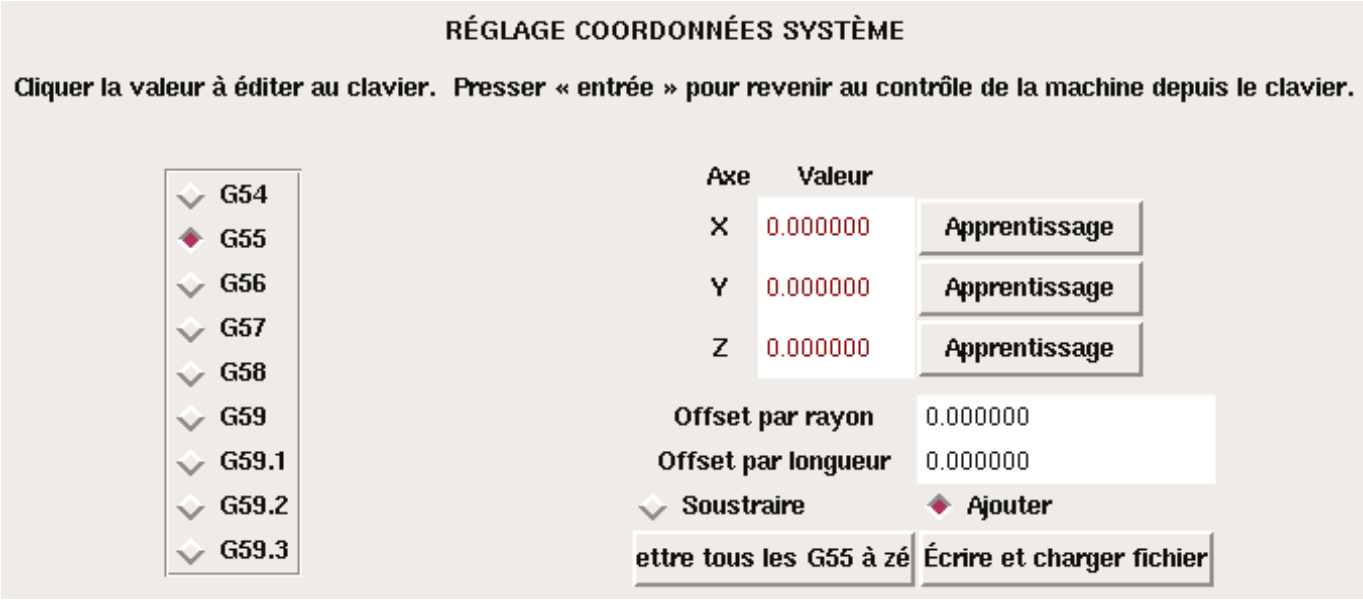



FIGURE 7.7 – Mini Offset Display

Il est également possible d'utiliser une pinnule puis d'ajouter le rayon et la longueur par le bouton *Apprentissage*. Pour celà, il peur être nécessaire d'ajouter ou de soustraire le rayon de la pinnule selon la surface qui est touchée. C'est la case à cocher *Soustraire* ou *Ajouter* qui permet ce choix.

Le bouton *Tous les Gxx à zéro* enléve tous les décalages du système courant visibles sur l'écran mais ne modifie pas les variables dans le fichier tant que le bouton *_écrire et recharger* n'est pas pressé. Le bouton *écrire et recharger* est à presser pour valider et recharger tous les changements effectués sur les valeurs des systèmes de coordonnées.

## 7.7 Raccourcis clavier

Un certain nombre de raccourcis clavier utilisés avec TkLinuxCNC ont été conservés avec Mini. Quelques uns ont changé pour étendre leurs fonctions ou pour faciliter les opérations sur une machine utilisant cette interface. Certaines touches opérent de la même manière quelque soit le mode. D'autres change en fonction du mode de travail de LinuxCNC.

### 7.7.1 Touches courantes

**Pause**
: Bascule du correcteur de vitesse travail

**Echap**
: Abandonne le mouvement

**F1**
: Bascule l'état de l'Arrêt d'Urgence

**F2**
: Marche/Arrêt machine

**F3**
: Mode manuel

**F4**
: Mode Auto

**F5**
: Mode MDI

**F6**
Réinitialise l'interpréteur

Ces touches fonctionnent seulement pour une machine utilisant des entrées sorties auxiliaires

**F7**
Marche/Arrêt du brouillard

**F8**
Marche/Arrêt arrosage

**F9**
Marche/Arrêt broche sens horaire

**F10**
Marche/Arrêt broche sens anti-horaire

**F11**
Diminue la vitesse de broche

**F12**
Augmente la vitesse de broche

### 7.7.2 Mode Manuel

**1-9 0**
Positionne le correcteur de vitesse d'avance travail de 10% en 10%, de 0 à 100%

**~**
Positionne le correcteur de vitesse à 0

**x**
Sélectionne l'axe X

**y**
Sélectionne l'axe Y

**z**
Sélectionne l'axe Z

**a**
Sélectionne l'axe A

**b**
Sélectionne l'axe B

**c**
Sélectionne l'axe C

**Gauche Droite**
jog de l'axe X

**Haut Bas**
jog de l'axe Y

**Pg.préc Pg.suiv**
jog de l'axe Z

- _:: jog de l'axe actif dans la direction moins

**+ =**
jog l'axe actif dans la direction plus

**Origine**
POM de l'axe actif

**i I**
Bascule cyclique des incréments de jog

Les touches suivantes ne fonctionnent que sur les machines utilisant des entrées sorties auxiliaires

**b**
Relâcher le frein de broche

**Alt-b**
Engager le frein de broche

### 7.7.3 Mode Auto

**1-9,0**
Positionne le correcteur de vitesse d'avance travail de 10% en 10%, de 0 à 100%

**~**
Positionne le correcteur de vitesse à 0

**o/O**
Ouvre un programme

**r/R**
Lance le programme ouvert

**p/P**
Met le programme en pause

**s/S**
Reprise d'un programme en pause

**a/A**
Avance d'une ligne dans un programme en pause

## 7.8 Divers

Une des possibilités de Mini est d'afficher n'importe quel axe, au delà de 2, comme étant un axe rotatif, il affiche alors des degrés pour celui-ci. Il converti également les unités en degrés quand un axe rotatif a le focus.

# Chapitre 8

# L'interface textuelle KEYSTICK

## 8.1 Introduction

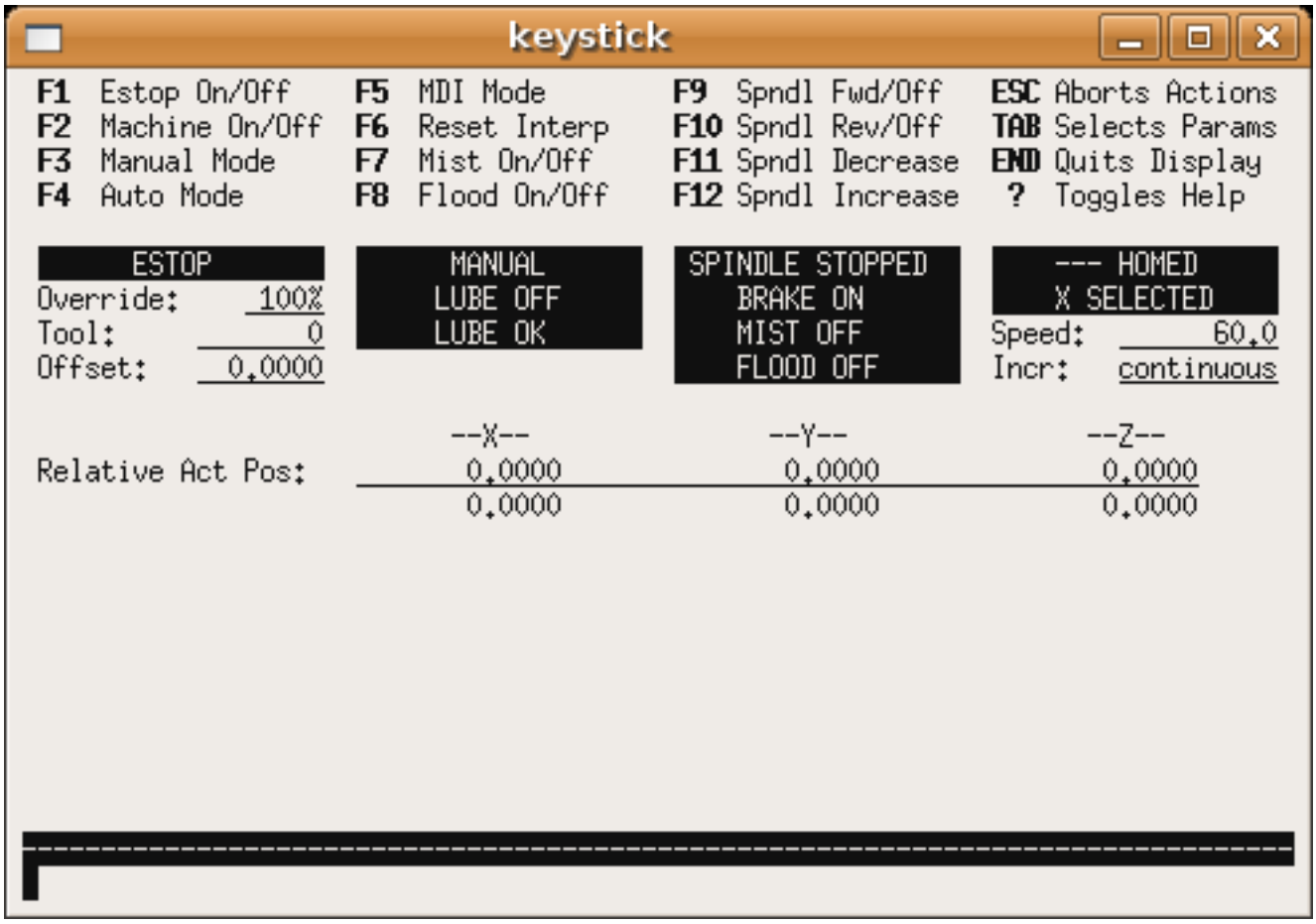



FIGURE 8.1 – L'interface utilisateur minimale

Keystick est une interface minimaliste textuelle.

## 8.2 Installation

Pour utiliser «keystick» modifier la section DISPLAY du fichier ini de la manière suivante:

```
[DISPLAY]

DISPLAY = keystick
```

## 8.3 Utilisation

Keystick est extrêmement simple à utiliser. Dans le mode MDI vous pouvez directement taper le G-code et il sera visible dans la partie supérieur. La touche «?» bascule avec ou sans aide.

[1]

1. ndt: keystick a été conçu pour être minimaliste, par conséquent son interface n'est pas traductible.

# Deuxième partie

# L'utilisation de LinuxCNC

# Chapitre 9

# Concepts importants pour l'utilisateur

## 9.1 La configuration machine

Le dessin suivant montre les directions de déplacement de l'outil et la position des fins de course de limite sur une fraiseuse classique. Noter le diagramme cartésien représentant les directions de déplacement de l'outil (Tool Direction). La direction de déplacement de la table et en opposition du système de coordonnées cartésiennes. Le système de coordonnées cartésiennes représente le sens de déplacement de l'outil. C'est toujours les déplacements de l'outil qui doivent être programmés pour que l'outil se déplace dans les directions correctes par rapport au matériel.

Noter également la position des fins de course et le sens d'activation de leurs cames. Plusieurs combinaisons sont possibles, par exemple il est possible, à l'inverse du dessin, de placer un seul fin de course fixe au milieu de la table et deux cames mobiles pour l'actionner. Dans ce cas les limites seront inversées, +X sera à droite de la table et -X à gauche. Cette inversion ne change rien du point de vue du sens de déplacement de l'outil.

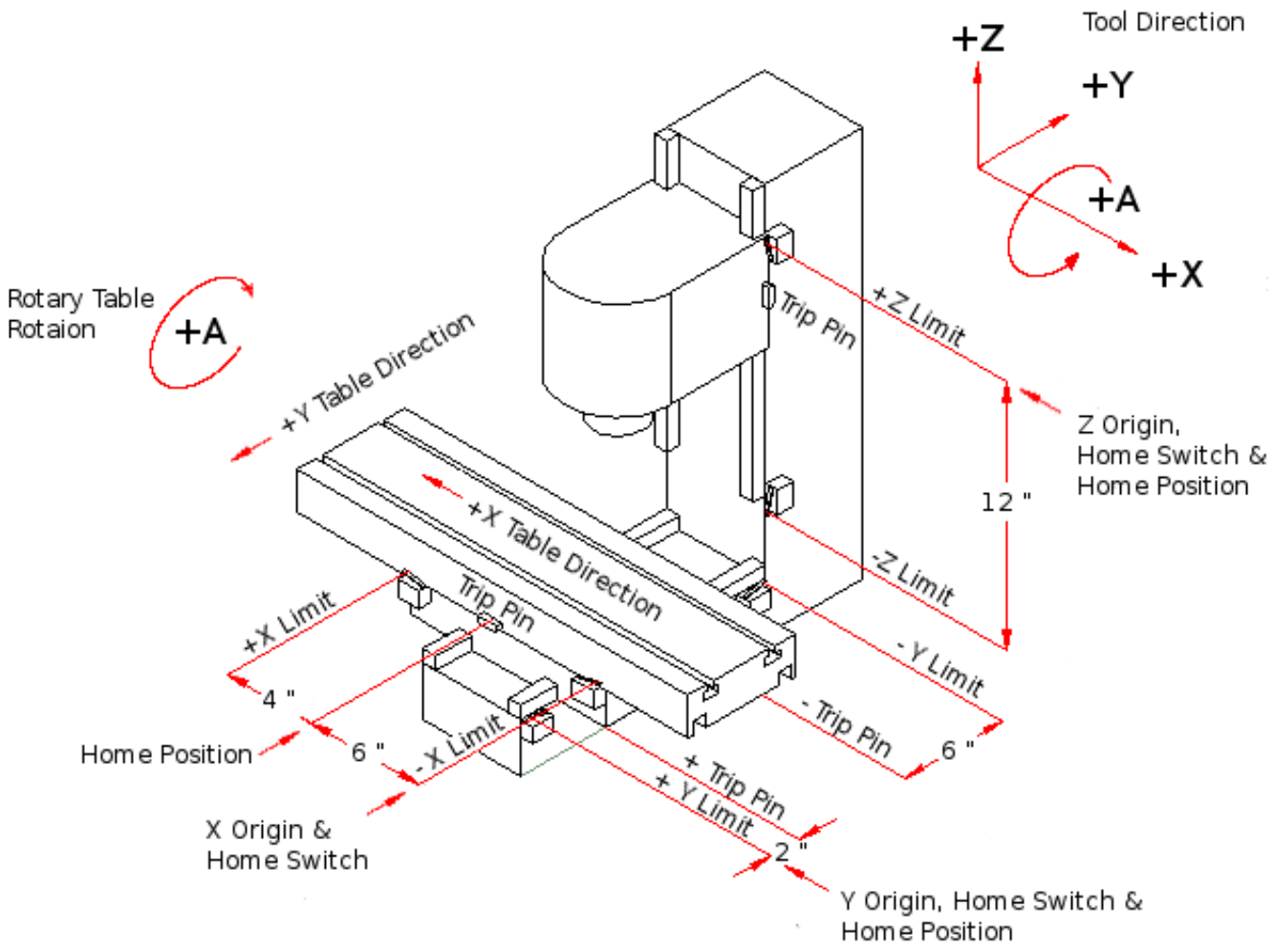



FIGURE 9.1 – Configuration typique d'une fraiseuse

Le dessin suivant montre les directions de déplacement de l'outil et la position des fins de course de limite sur un tour classique.

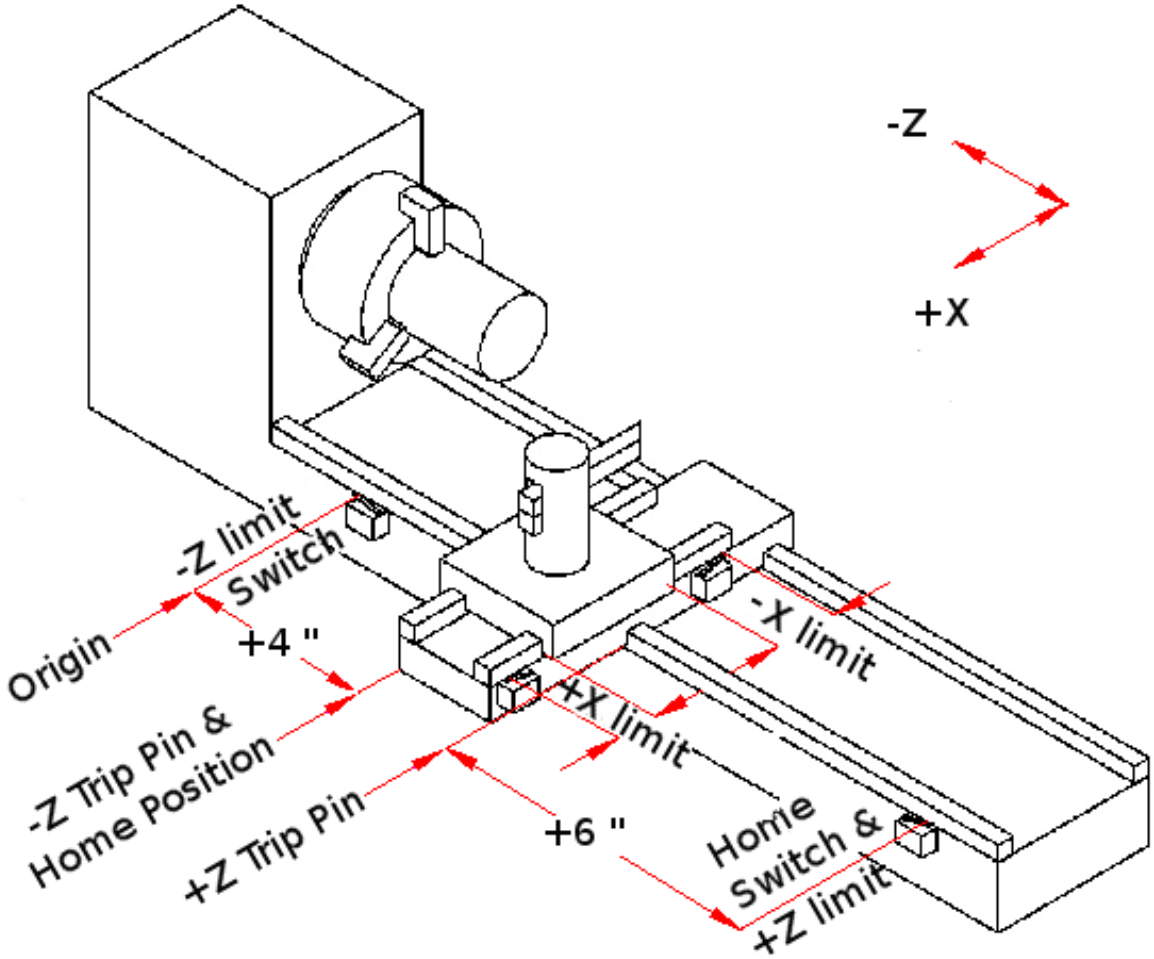



FIGURE 9.2 – Configuration typique d'un tour

## 9.2 Contrôle de trajectoire

### 9.2.1 La planification de trajectoire

La planification de trajectoire est en général, le moyen qui permet à LinuxCNC de suivre le chemin spécifié par le programme G-code, tout en restant dans les limites permises par la machine.

Un programme en G-code ne peut jamais être exactement suivi. Par exemple imaginez que vous spécifiez dans une ligne du programme les mouvements suivants:

```
G1 X10 F100 (G1 un mouvement élinaire, X10 la destination, F100 la vitesse)
```

En réalité, la totalité du mouvement ne peut pas être effectuée à F100, puisque la machine commence le mouvement à une vitesse nulle, elle doit accélérer pour se déplacer vers X=10, puis décélérer pour revenir à une vitesse nulle en fin de mouvement. Parfois une portion du mouvement se fera bien à F100, mais pour beaucoup de mouvements, spécialement les petits mouvements, la vitesse spécifiée ne sera jamais atteinte.

Les accélérations et décélérations de base décrite ici ne sont pas complexes et ne nécessite pas de compromis. Les contraintes des axes de la machine sont placés dans le fichier INI, comme la vitesse maximum de l'axe et l'accélération ne devant pas être dépassées par le planificateur de trajectoire.

### 9.2.2 Le suivi du parcours

Un problème plus compliqué est posé par le suivi du parcours. Quand vous programmez un angle droit en G-code, le planificateur de trajectoire peut suivre différents parcours, tous sont bons dans certains cas; il peut décélérer et s'arrêter exactement sur les coordonnées du sommet de l'angle, puis accélérer dans la direction perpendiculaire. Il peut également faire ce qui est appelé le mode *trajectoire continue*, qui consiste à maintenir la vitesse d'avance en passant vers le sommet de l'angle, ce qui nécessite d'arrondir l'angle de façon à respecter les contraintes machine. Vous pouvez remarquer qu'il y a dans ce cas un compromis: vous pouvez ralentir pour avoir un meilleur suivi du parcours, ou conserver une vitesse d'avance élevée au détriment de la finesse des angles, du fait d'un moins bon suivi du parcours. Selon les particularités de l'usinage, du matériau, de l'outillage, etc., le programmeur devra décider du bon compromis.

### 9.2.3 La programmation du planificateur

Les commandes de contrôle de trajectoire sont les suivantes:

**G61**
: (mode trajectoire exacte) G61 indique au planificateur de suivre exactement la trajectoire prévue.

**G61.1**
: (mode Arrêt exact) G61.1 demande au planificateur de s'arrêter exactement à la fin de chaque segment. Le parcours sera suivi avec exactitude mais les arrêts complets de l'avance peuvent se révéler destructeurs pour la pièce ou l'outillage, selon les particularités de l'usinage.

**G64**
: (mode trajectoire continue sans tolérance) Le mode G64 est le mode par défaut au démarrage de LinuxCNC. G64 est juste une trajectoire continue, le *Détecteur naive CAM* n'est pas activé. G64 et G64 P0 indiquent au planificateur de sacrifier la précision de suivi du parcours pour conserver une vitesse d'avance élevée. Ce mode est nécessaire pour certains types de matériaux ou d'outillages pour lesquels l'arrêt exact est dangereux. Il peut très bien fonctionner tant que le programmeur garde à l'esprit que le parcours d'outil pourra être plus arrondi que celui indiqué par le programme. Dans le cas d'un mouvement en G0 (rapide) avec G64, faire preuve de prudence sur les mouvements de dégagement et prévoir suffisamment de distance pour éviter les obstacles selon les capacités d'accélérations de la machine.

**G64 Px.xxx**
: (mode trajectoire continue avec tolérance) Ce mode active le *Détecteur naive CAM*" et active le mode trajectoire continue avec tolérance. Si vous utilisez le millimètre comme unité et programmez G64 P1.27, vous dites au planificateur que vous souhaitez une vitesse d'avance continue, mais qu'aux coins programmés vous voulez un ralentissement suffisant pour que le parcours de l'outil puisse rester à moins de 1.27mm du parcours programmé. L'amplitude exacte du ralentissement dépend de la géométrie de l'angle programmé et des contraintes machine, mais la seule chose dont le programmeur ait à se soucier est la tolérance, ce qui lui donne le contrôle complet des compromis du suivi de parcours. La tolérance de ce mode peut être modifiée tout au long du programme si nécessaire. Attention: spécifier un G64 P0 aura le même effet qu'un G64 seul (voir ci-dessus), c'est rendu nécessaire pour conserver la compatibilité ascendante avec les anciens programmes G-code. Voir le chapitre sur le G-code pour plus d'information sur G64 P- Q-.

**Trajectoire continue sans tolérance**
: Le point contrôlé touchera chaque mouvement spécifié à au moins un point. La machine ne pourra jamais se déplacer à une vitesse d'avance telle qu'elle ne puisse pas s'arrêter avec précision à la fin du mouvement en cours (ou du prochain mouvement, si vous mettez en pause lorsque la trajectoire est déjà commencée). La distance avec le point final du mouvement est aussi grande que nécessaire pour maintenir la meilleure vitesse d'avance possible pendant le parcours.

**Détecteur Naive Cam**
: Les mouvements successifs en G1, concernant uniquement les axes XYZ, dont la déviation par rapport à une ligne droite est inférieur à P, sont fusionnés en une seule ligne droite. Ce mouvement fusionné remplace les mouvements individuels en G1 pour obtenir une nouvelle trajectoire avec tolérance. Entre les mouvements successifs, le point contrôlé ne passera jamais à plus de P- du point final du mouvement en cours. Le point contrôlé touchera au moins un point de chacun des mouvements. La machine ne pourra jamais se déplacer à une vitesse ne lui permettant pas de venir s'arrêter exactement à la fin du mouvement actuel (ou du prochain mouvement, si vous mettez en pause lorsque la trajectoire est déjà commencée). En mouvement G2/3 dans le plan G17 (XY) quand la déviation maximale entre un arc et une ligne droite est plus petite que la tolérance G64 Q- l'arc est brisé en deux lignes (du début de l'arc à son milieu et du milieu à la fin de l'arc). Ces deux tronçons sont ensuite soumis à l'algorithme Naïve cam des lignes. Ainsi, les cas ligne-arc, arc-arc et arc-ligne, comme les cas ligne-ligne bénéficient du traitement *Détecteur naive CAM*. Les performances de contourage sont accrues grâce à la simplification de la trajectoire.

Dans la figure suivante la ligne bleue représente la vitesse machine actuelle. La ligne rouge représente la capacité d'accélération de la machine. La ligne horizontale sous chaque tracé est le mouvement planifié. Le tracé supérieur montre comment le planificateur de trajectoire ralenti la machine quand des petits mouvements sont rencontrés. Ceci pour rester dans les limites fixées par les paramètres d'accélération de la machine et être capable de s'arrêter exactement à la fin du prochain mouvement. Le tracé du bas montre l'effet du détecteur Naive Cam pour combiner les mouvements et fournir une amélioration conséquente dans le suivi de la vitesse programmée.

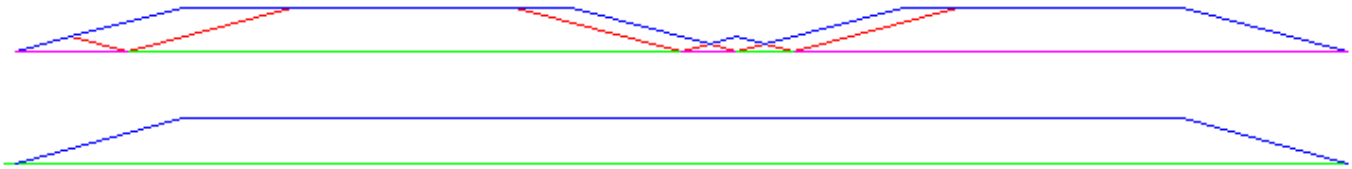


FIGURE 9.3 – Détecteur Naive Cam

### 9.2.4 Planification des mouvements

Assurez-vous que les mouvements soient *assez longs* pour convenir à votre machine/matériel. Principalement en raison de la règle selon laquelle "la machine ne pourra jamais se déplacer à une vitesse ne lui permettant pas de venir s'arrêter complètement à la fin du mouvement actuel", il y a une longueur minimale de déplacement permettant à la machine d'atteindre la vitesse demandée avec un réglage d'accélération donné.

Les phases d'accélération et de décélération utilisent chacune la moitié de la variable MAX_ACCELERATION du fichier .ini. Avec une trajectoire continue c'est exactement inversé, ce qui fait que l'accélération totale de l'axe est égal à la variable MAX_ACCELERATION. Dans d'autres cas, l'accélération actuelle de la machine est un peu inférieure à celle du fichier ini.

Pour maintenir la vitesse d'avance, le mouvement doit être plus long que la distance qui lui est nécessaire pour accélérer de zéro à la vitesse souhaitée, puis de décélérer pour s'arrêter. En utilisant $A$ comme étant *1/2* de la variable MAX_ACCELERATION du fichier ini et $F$ comme étant la vitesse d'avance *en unités par seconde*, le temps d'accélération sera $t_a = F/A$ et la distance d'accélération sera $d_a = F*t_a/2$. Les temps et distance de décélération sont les mêmes, ce qui fait que la distance critique $d = d_a + d_d = 2 * d_a = F^2/A$.

Par exemple, pour une vitesse d'avance de *25mm par seconde* et une accélération de *250 mm/sec*$^2$, la distance critique sera de $10^2/100 = 100/100 = 1mm$. Pour une vitesse d'avance de *5mm par seconde*, la distance critique ne serait que de $5^2/100 = 25/100 = 0.25mm$.

## 9.3 G-code

### 9.3.1 Par défaut

Quand LinuxCNC démarre pour la première fois beaucoup de G et M codes sont chargés par défaut. Les codes actifs courants sont visibles dans l'interface Axis, dans l'onglet *Données manuelles* dans le champ *G-codes actifs* Ces codes G et M définissent le comportement de LinuxCNC et il est important de bien comprendre la signification de chacun avant de démarrer LinuxCNC. Ces codes par défaut peuvent être modifiés lors du lancement d'un fichier de G-codes puis laissés dans différents états qui seront identiques lors d'une nouvelle session de LinuxCNC. La bonne pratique consiste à mettre dans le préambule de chaque fichier de G-codes les codes nécessaires pour le travail demandé et ne pas supposer que ceux par défaut conviendront. Imprimer la page des références rapides du G-code peut aider à se rappeler la signification de chacun d'eux.

## 9.4 Vitesse d'avance

Si vous avez un tour ou un axe rotatif, pour savoir comment la vitesse d'avance s'applique selon que l'axe est linéaire ou rotatif, lire et comprendre la section vitesse d'avance du manuel de l'utilisateur.

## 9.5 Compensation de rayon d'outil

La compensation de rayon d'outil (G41/G42) nécessite que l'outil puisse usiner tout au long de la trajectoire programmée sans interférer avec les mouvements d'entrée ou de sortie. Si c'est impossible avec le diamètre de l'outil courant, une erreur est signalée. Un diamètre d'outil inférieur est peut être utilisable sans erreur pour le même parcours. Ce qui signifie que quand ce type de problème se présente, il est possible de programmer un outil plus petit pour usiner le même parcours sans erreur. Voir la section compensation de rayon d'outil pour plus d'informations.

## 9.6 Prise d'origine machine

Après le démarrage de LinuxCNC chaque axe doit être référencé sur son point d'origine machine avant tout mouvement ou commande MDI.

Pour déroger à ce comportement par défaut, ou pour utiliser l'interface Mini, il est possible d'ajuster l'option NO_FORCE_HOMING = 1 dans la section [TRAJ] du fichier ini.

## 9.7 Changement d'outil

Il existe plusieurs options pour effectuer un changement d'outil. Voir la section [EMCIO] dans le manuel de l'intégrateur pour les informations sur la configuration de ces options. Voir également les sections G28 et G30 du manuel de l'utilisateur.

## 9.8 Systèmes de coordonnées

Les systèmes de coordonnées peuvent être déroutant au premier abord. Avant de démarrer une machine CNC, il est important de bien comprendre les bases des systèmes utilisés par LinuxCNC. Pour explorer plus en profondeur les systèmes de coordonnées utilisés par LinuxCNC, voir la section xxxxx de ce manuel.

### 9.8.1 G53 Coordonnées machine

Quand vous réalisez une prise d'origine de plusieurs axes de LinuxCNC, vous passez G53, les coordonnées système, à 0 pour chacun des axes concernés.
– La prises d'origine ne modifient en rien les autres systèmes de coordonnées, ni les compensations d'outil.
La seule façon de se déplacer en mode G53, en coordonnées machine, c'est de programmer un G53 sur la même ligne que celle d'un mouvement. En fonctionnement normal, vous êtes dans le système de coordonnées G54.

### 9.8.2 G54 à 59.3 Coordonnées utilisateur

Normalement vous utilisez le système de coordonnées G54. Quand un décalage est appliqué au système de coordonnées utilisateur courant, dans Axis, une petite sphère bleue avec des rayons est affichée à l'emplacement de l'origine machine quand la visu affiche *Position: Relative Actuelle*. Si votre décalage utilise temporairement les coordonnées machine, depuis le menu Machine ou en programmant *G10 L2 P1 X0 Y0 Z0* à la fin du programme G-Code. Modifiez la valeur du mot *P* en fonction du système de coordonnées dont vous voulez effacer le décalage.
– Les décalages stockés dans un système de coordonnées utilisateur sont conservés à l'arrêt de LinuxCNC.
– Dans Axis, utiliser le bouton *Toucher* décalera le système de coordonnées utilisateur choisi.

### 9.8.3 Quand vous êtes perdu

Si vous avez des difficultés pour obtenir 0,0,0 sur la visu alors que vous pensez que vous devriez l'avoir, c'est peut être provoqué par plusieurs décalages programmés et qu'il conviendrait de supprimer. Pour cela:
– Placez vous sur l'origine machine avec *G53 G0 X0 Y0 Z0*
– Supprimez tous les décalages *G92* avec *G92.1*
– Utilisez les coordonnées utilisateur avec *G54*
– Rendez les coordonnées utilisateur *G54*, identiques aux coordonnées machine avec *G10 L2 P1 X0 Y0 Z0 R0*
– Annulez les offsets d'outil avec *G49*
– Activez l'affichage des coordonnées relatives depuis le menu.
Maintenant vous devriez être, à l'origine machine *X0 Y0 Z0* et le système de coordonnées relatives devrait être le même que le système de coordonnées machine. :lang: fr :toc:

# Chapitre 10

# Aperçu global d'une machine CNC

Cette section donne une description des différents organes constituant une machine à commande numérique (CNC).

## 10.1 Composants mécaniques

Une machine à commande numérique dispose de beaucoup de composants mécaniques pouvant être contrôlés, ou qui peuvent avoir une incidence sur la façon dont le contrôle de la machine s'effectue. Cette section décrit les composants qui interagissent avec l'interpréteur. Les autres composants mécaniques, comme les boutons de jog, ne seront pas décrits ici, même si ils affectent le contrôle.

### 10.1.1 Axes

Toute machine à commande numérique dispose d'un ou de plusieurs axes. Les différents types de machines ont différentes combinaisons d'axes. Par exemple, une fraiseuse 4 axes peut avoir la combinaison d'axes XYZA ou XYZB. Un tour classique aura les axes XZ. Une machine de découpe à fil chaud aura les axes XYUV. [1] [2]

#### 10.1.1.1 Axes linéaires primaires

Les axes X, Y et Z produisent des mouvements linéaires dans trois directions, mutuellement orthogonales.

#### 10.1.1.2 Axes linéaires secondaires

Les axes U, V et W produisent des mouvements linéaires dans trois directions, mutuellement orthogonales. Habituellement, X et U sont parallèles, Y et V sont parallèles et Z et W sont parallèles.

#### 10.1.1.3 Axes rotatifs

Les axes A, B et C produisent des mouvements angulaires (rotations). Habituellement, l'axe de rotation de A est parallèle à X, l'axe de rotation de B est parallèle à Y et l'axe de rotation de C est parallèle à Z.

1. Si le mouvement des composants mécaniques n'est pas indépendant, comme sur une machine hexapode, le langage RS274/NGC et les fonctions standards seront quand même utilisables, tant que le contrôle de bas niveau sait comment contrôler les mécanismes actuels pour produire le mouvement relatif de l'outil et de la pièce qui auraient été produits par des axes indépendants. C'est appelé, la cinématique.
2. Avec LinuxCNC, le cas de la machine à portique XYYZ avec deux moteurs pour un axe est mieux traité par la cinématique que par un axe linéaire supplémentaire.

### 10.1.2 Broche

Une machine à commande numérique est équipée d'une broche qui maintient un outil coupant, un palpeur ou d'autres outils. La broche peut tourner dans les deux sens. Elle peut être conçue pour tourner à vitesse constante mais réglable. Excepté sur les machines dont la broche est montée sur un axe rotatif, l'axe de la broche est maintenu parallèle à l'axe Z et il est coïncident avec l'axe Z quand X et Y sont à zéro. La broche peut être stoppée sur une position fixée ou non.

### 10.1.3 Arrosages

Une machine à commande numérique peut être équipée d'un système fournissant l'arrosage fluide ou un arrosage par gouttelettes.

### 10.1.4 Correcteurs de vitesse d'avance et de broche

Une machine à commande numérique est équipée de boutons de réglage de la vitesse d'avance et de la vitesse de rotation de la broche, ils laissent l'opérateur corriger les vitesses nécessaires pour la broche et l'avance travail, il peut ainsi augmenter ou réduire les vitesses programmées.

### 10.1.5 Bouton d'effacement de bloc

Une machine à commande numérique peut être équipée d'un bouton d'effacement de bloc. Voir la section effacement de bloc.

### 10.1.6 Bouton d'arrêt optionnel du programme

Une machine à commande numérique peut être équipée d'un bouton d'arrêt du programme. Voir la section arrêts optionnels.

## 10.2 Composants de contrôle et de données

### 10.2.1 Axes linéaires

Les axes X, Y et Z forment un système de coordonnées orthogonales standard. La position d'un axe s'exprime en utilisant ses coordonnées.

### 10.2.2 Axes linéaires secondaires

Les axes U, V et W forment également un système de coordonnées standard. X et U sont parallèles, Y et V sont parallèles enfin Z et W sont parallèles.

### 10.2.3 Axes rotatifs

Les axes rotatifs se mesurent en degrés. Leur sens de rotation positif est le sens anti-horaire quand l'observateur est placé face à l'axe. [3]

### 10.2.4 Point contrôlé

Le point contrôlé est le point dont la position et la vitesse de déplacement sont contrôlés. Quand la compensation de longueur d'outil est à zéro (valeur par défaut), c'est un point situé sur l'axe de la broche et proche de la fin de celle-ci. Cette position peut être déplacée le long de l'axe de la broche en spécifiant une compensation de longueur d'outil. Cette compensation correspond généralement à la longueur de l'outil coupant courant. Ainsi, le point contrôlé est à la pointe de l'outil. Sur un tour, les correcteurs d'outil peuvent être spécifiés pour les axes X et Z, le point contrôlé est à la pointe de l'outil ou (correction du rayon de bec) légèrement en retrait du point d'intersection des droites perpendiculaires formées par l'axe des points de tangence à la pièce, de face et sur le côté de l'outil.

3. Si les parallélismes sont particuliers, le constructeur du système devra indiquer à quels sens de rotation correspondent horaire et anti-horaire.

### 10.2.5 Mouvement linéaire coordonné

Pour mener un outil sur une trajectoire spécifiée, une machine à commande numérique doit coordonner les mouvements de plusieurs axes. Nous utilisons le terme *mouvement linéaire coordonné* pour décrire une situation dans laquelle, nominalement, chacun des axes se déplace à vitesse constante et tous les axes se déplacent de leur point de départ à leur point d'arrivée en même temps. Si deux des axes X, Y, Z (ou les trois) se déplacent, ceci produit un mouvement en ligne droite, d'où le mot *linéaire* dans le terme. Dans les véritables mouvements, ce n'est souvent pas possible de maintenir la vitesse constante à cause des accélérations et décélérations nécessaires en début et fin de mouvement. C'est faisable, cependant, de contrôler les axes ainsi, chaque axe doit en permanence faire la même fraction du mouvement requis que les autres axes. Ceci déplace l'outil le long du même parcours et nous appelons aussi ce genre de mouvement, mouvement linéaire coordonné.

Un mouvement linéaire coordonné peut être exécuté soit en vitesse travail, soit en vitesse rapide, ou il peut être synchronisé à la rotation de la broche. Si les limites physiques de l'axe rendent le déplacement impossible, tous les axes seront ralentis pour maintenir le parcours prévu.

### 10.2.6 Vitesse d'avance

La vitesse à laquelle le point contrôlé se déplace est ajustable par l'opérateur. Sauf cas particulier, vitesse inverse du temps, vitesse par tour, voir la section sur les modes de vitesse, dans l'interpréteur, l'interprétation des vitesses est la suivante:

1. Si le déplacement concerne un des axes XYZ, F est en unités machine par minute dans le système Cartésien XYZ et les mouvements des autres axes (UVWABC) sont également dans un même mode de coordonnées.
2. Autrement, si le déplacement concerne un des axes UVW, F est en unités machine par minute dans le système Cartésien UVW, tous les autres axes (ABC) se déplacent dans un même mode de coordonnées.
3. Autrement, le mouvement est purement rotatif et le mot F est en unités de rotation dans le système pseudo-Cartésien ABC.

### 10.2.7 Arrosage

Arrosage fluide ou par gouttelettes peuvent être activés séparément. Le langage RS274/NGC les arrête ensemble, voir la section des contrôles d'arrosage.

### 10.2.8 Temporisation

Une temporisation peut être commandée (ex: pour immobiliser tous les axes) pendant une durée spécifique. La broche n'est pas arrêtée pendant une temporisation! Sans s'occuper du mode de contrôle de trajectoire la machine s'arrêtera exactement à la fin du dernier mouvement avant la temporisation.

### 10.2.9 Unités

Les unités utilisées pour les distances le long des axes X, Y et Z peuvent être les pouces ou les millimètres. La vitesse de rotation de la broche est en tours par minute. Les positions des axes rotatifs sont exprimées en degrés. Les vitesses d'avance sont exprimées en unités machine par minute ou en degrés par minute ou en unités de longueur par tour de broche, comme décrit dans la section des vitesses.

### 10.2.10 Position courante

Le point contrôlé est toujours à un emplacement appelé la *position courante*, et le contrôleur sait toujours où est cette position. Les valeurs représentant la position courante doivent être ajustées en l'absence de tout mouvement des axes si un de ces événements a lieu:

1. Les unités de longueur ont changé.
2. La compensation de longueur d'outil a changé.
3. Le décalage d'origine a changé.

### 10.2.11 Choix du plan de travail

Il y a toujours un plan sélectionné, qui doit être le plan XY, le plan YZ, ou le plan XZ de la machine. L'axe Z est, bien sûr, perpendiculaire au plan XY, l'axe X perpendiculaire au plan YZ et l'axe Y perpendiculaire au plan XZ.

### 10.2.12 carrousel d'outils

Aucun ou un outil est assigné à chaque emplacement dans le carrousel.

### 10.2.13 Changeur d'outil

Une machine à commande numérique peut commander un changeur d'outils.

### 10.2.14 Chargeur de pièce

Les deux porte-pièces peuvent être intervertis par commande.

### 10.2.15 Chargeur de pièces

Une machine à commande numérique peut être équipée d'un système de chargement des pièces. Le système se compose de deux porte-pièces sur lesquels sont fixés les bruts des pièces à usiner. Un seul porte-pièce à la fois est en position d'usinage.

### 10.2.16 Boutons des correcteurs de vitesses

Les boutons des correcteurs de vitesses peuvent être activés (ils fonctionnent normalement) ou rendus inopérants (Ils n'ont plus aucun effet). Le langage RS274/NGC dispose d'une commande qui active tous les boutons et une autre qui les désactive. Voir l'inhibition et l'activation des correcteurs de vitesse. Voir également ici pour d'autres détails.

### 10.2.17 Modes de contrôle de trajectoire

La machine peut être placée dans un de ces trois modes de contrôle de trajectoire:
- mode arrêt exact:: En mode arrêt exact, le mobile s'arrête brièvement à la fin de chaque mouvement programmé.
- mode trajectoire exacte:: En mode trajectoire exacte, le mobile suit la trajectoire programmée aussi précisément que possible, ralentissant ou s'arrêtant si nécessaire aux angles vifs du parcours.
- mode trajectoire continue avec tolérance optionnelle:: En mode trajectoire continue, les angles vifs du parcours peuvent être légèrement arrondis pour que la vitesse soit maintenue (sans dépasser la tolérance, si elle est spécifiée).

Voir également les G-codes G61/G61.1 et G64 des contrôles de trajectoire.

## 10.3 Interaction de l'interpréteur avec les boutons

L'interpréteur interagit avec plusieurs boutons de commande. Cette section décrit ces interactions plus en détail. En aucun cas l'interpréteur ne connait ce que sont les réglages de ces boutons.

### 10.3.1 Boutons de correction de vitesses

L'interpréteur de commande RS274/NGC autorise (M48) ou interdit (M49) l'action des boutons d'ajustement des vitesses. Pour certains mouvements, tels que la sortie de filet à la fin d'un cycle de filetage, les boutons sont neutralisés automatiquement.

LinuxCNC réagit aux réglages de ces boutons seulement quand ils sont autorisés.

### 10.3.2 Bouton d'effacement de bloc

Si le bouton *Effacement de bloc* est actif, les lignes de code RS274/NGC commençant par le caractère barre de fraction (caractère d'effacement de bloc) ne sont pas interprétées. Si le bouton est désactivé, ces mêmes lignes sont interprétées. Normalement le bouton d'effacement de bloc doit être positionné avant de lancer le programme G-code.

### 10.3.3 Bouton d'arrêt optionnel du programme

Si ce bouton est actif et qu'un code M1 est rencontré, le programme est mis en pause.

== Fichier d'outils

Un fichier d'outils est requis par l'interpréteur. Le fichier indique dans quels emplacements du carrousel sont placés les outils, la longueur et le diamètre de chacun des outils. Le nom de la table d'outils est défini sous cette forme dans le fichier ini:

```
[EMCIO]

# tool table file
TOOL_TABLE = tooltable.tbl
```

Il est également possible de donner à la table d'outils le même nom que le fichier ini, mais avec une extension tbl, par exemple:

```
TOOL_TABLE = acme_300.tbl
```

ou encore:

```
TOOL_TABLE = EMC-AXIS-SIM.tbl
```

D'autres informations sont disponibles sur les spécificités du format de la table d'outils.

== Paramètres

Dans le langage RS274/NGC, la machine maintient un tableau de 5400 paramètres numériques. La plupart d'entre eux ont un usage spécifique. Le tableau de paramètres est persistant, même quand la machine est mise hors tension. LinuxCNC utilise un fichier de paramètres et assure sa persistance, il donne à l'interpréteur la responsabilité d'actualiser le fichier. L'interpréteur lit le fichier quand il démarre et l'écrit juste avant de s'arrêter.

Tous les paramètres sont disponibles pour une utilisation dans les programmes de G-code.

Un fichier de paramètres est composé d'un certain nombre de lignes d'en-tête, suivies par une ligne vide, suivie d'un nombre quelconque de lignes de données. Les lignes d'en-tête sont ignorées par l'interpréteur. Il est important qu'il y ait une ligne vide (sans espace ni tabulation), avant les données. La ligne d'en-tête montrée dans le tableau ci-dessous, décrit les colonnes de données, il est donc proposé (mais pas obligatoire) que cette ligne soit toujours présente.

L'interpréteur lit seulement les deux premières colonnes du tableau. Il ignore la troisième colonne, Commentaire.

Chaque ligne du fichier contient le numéro d'index d'un paramètre dans la première colonne et la valeur attribuée à ce paramètre, dans la deuxième colonne. La valeur est représentée par une nombre flottant en double précision à l'intérieur de l'interpréteur, mais le point décimal n'est pas exigé dans le fichier. Le format des paramètres décrit ci-dessous, est obligatoire et doit être utilisé pour tous les fichiers de paramètres, à l'exception des paramètres représentant une valeur sur un axe rotatif inutilisé, qui peuvent être omis. Une erreur sera signalée si un paramètre requis est absent. Un fichier de paramètres peut inclure tout autre paramètre, tant que son numéro est compris dans une fourchette de 1 à 5400. Les numéros de paramètre doivent être disposés dans l'ordre croissant. Sinon, une erreur sera signalée. Le fichier original est copié comme fichier de sauvegarde lorsque le nouveau fichier est écrit. Les commentaires ne sont pas conservés lorsque le fichier est écrit.

TABLE 10.1: Format d'un fichier de paramètres

| Numéro d'index | Valeur | Commentaire |
|---|---|---|
| 5161 | 0.0 | G28 pom X |
| 5162 | 0.0 | G28 pom Y |

# Chapitre 11

# Systèmes de coordonnées et décalages

## 11.1 Introduction

Dans ce chapitre, nous allons tenter de démystifier les systèmes de coordonnées. C'est une notion capitale pour maitriser le fonctionnement d'une machine CNC, sa configuration et son utilisation.

Nous montrerons également, qu'il est très intéressant d'utiliser un point de référence sur le brut ou la pièce et de faire travailler le programme à partir de ce point, sans avoir à tenir compte d'où est placée la pièce sur la machine pendant l'usinage. Ce chapitre va donc introduire les décalages et comment ils sont utilisés par LinuxCNC:
– Les coordonnées machine.(G53)
– Les neuf décalages d'origine pièce.(G54 à G59.3)
– Un jeu de décalages globaux.(G92)

## 11.2 Commande en coordonnées machine: G53

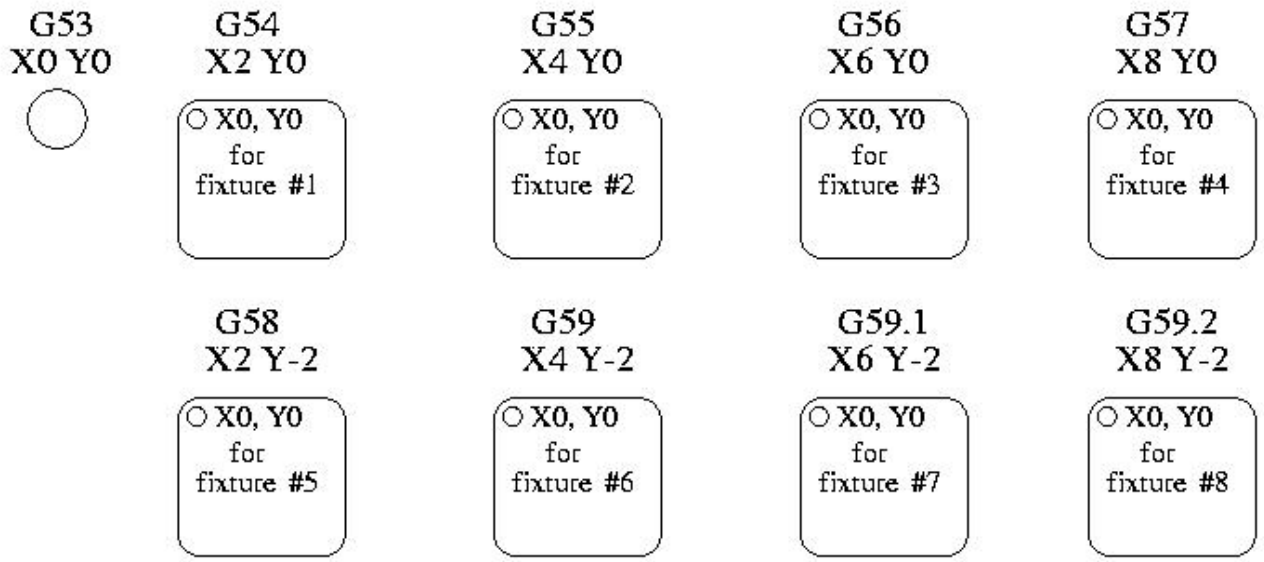
Indépendamment de tout décalage pouvant être actif, un G53 dans une ligne de code indique à l'interpréteur de se déplacer aux positions réelles des axes (positions absolues), commandées dans la ligne. Par exemple:

```
G53 G0 X0 Y0 Z0
```

déplacera le mobile depuis la position actuelle vers la position où les coordonnées machine des trois axes seront à zéro. Vous pouvez utiliser cette commande si vous avez une position fixe pour le changement d'outil ou si votre machine dispose d'un changeur automatique. Vous pouvez aussi utiliser cette commande pour dégager la zone de travail et accéder à la pièce dans l'étau.

G53 est une commande non modale. Elle doit être utilisée sur chaque ligne où un mouvement basé sur les coordonnées machine est souhaité.

## 11.3 Décalages pièce (G54 à G59.3)




**Exemple de décalages pour 8 ilots** Le décalage d'origine est utilisé pour séparer le point de référence de la pièce, de l'origine machine, créant ainsi un système de coordonnées (relatif), propre à chaque pièce et décalé du système de coordonnées machine (absolu). Il permet, entre autre, dans le cas de pièces multiples mais semblables, de créer en décalant ses origines, le système de coordonnées de chaque pièce, le programme restant le même. Un cas typique d'utilisation de cette fonctionnalité, pour usiner huit ilots identiques sur la même pièce, est illustré sur la figure ci-dessus.

Les valeurs des décalages sont enregistrées dans le fichier VAR qui est requis par le fichier INI durant le démarrage de LinuxCNC. Dans l'exemple ci-dessous, qui utilise G55, la valeur de chacun des axes de G55 est enregistrée dans une variable numérotée.

| Variable | Valeur |
|---|---|
| 5241 | 0.000000 |
| 5242 | 0.000000 |
| 5243 | 0.000000 |
| 5244 | 0.000000 |
| 5245 | 0.000000 |
| 5246 | 0.000000 |

Dans le schéma d'un fichier VAR, la première variable contient la valeur du décalage de l'axe X, la seconde variable le décalage de l'axe Y et ainsi de suite pour les six axes. Il y a une série de variables de ce genre pour chacun des décalages pièce.

Chacune des interfaces graphiques offre un moyen de fixer les valeurs de ces décalages. Vous pouvez également définir ces valeurs en modifiant le fichier VAR lui-même, puis faire un [reset], pour que LinuxCNC lise les nouvelles valeurs. Pour notre exemple, nous allons éditer directement le fichier pour que G55 prenne les valeurs suivantes:

| Variable | Valeur |
|---|---|
| 5241 | 2.000000 |
| 5242 | 1.000000 |
| 5243 | -2.000000 |
| 5244 | 0.000000 |
| 5245 | 0.000000 |
| 5246 | 0.000000 |

Vous pouvez voir que les positions zéro de G55 sont passées en X = 2, Y = 1, et Z = -2 éloignées donc de l'origine absolue (machine).

Une fois que les valeurs sont assignées, un appel de G55 dans une ligne de programme décalera le point de référence zéro, l'origine, vers les valeurs enregistrées. La ligne suivante décalera chacun des axes vers sa nouvelle position d'origine. Contrairement à G53, les commandes G54 à G59.3 sont modales. Elles agissent sur toutes les lignes de G-code du programme après qu'une ait été rencontrée. Voyons le programme qui pourrait être écrit à l'aide de la figure des décalages d'ilots, il suffira d'un seul point de référence pour chacune des pièces pour faire tout le travail. Le code suivant est proposé comme exemple pour faire un rectangle, il utilisera les décalages G55 que nous avons expliqué précédemment.

```
G55 G0 X0 Y0 Z0
G1 F2 Z-0.2000
X1
Y1
X0
Y0
G0 Z0
G54 X0 Y0 Z0
M2
```

*Mais,* dites vous, *pourquoi y a-t-il un G54 vers la fin ?* C'est une pratique courante de quitter le système de coordonnées G54 avec l'ensemble des valeurs d'axes à zéro afin de laisser un code modal basé sur les positions machine absolues. Nous le faisons avec cette commande qui met la machine à zéro. Il aurait été possible d'utiliser G53 et d'arriver au même endroit, mais la commande n'aurait pas été modale, les commandes suivantes auraient voulu retourner dans le système de coordonnées du G55 toujours actif.

```
G54     utilise les érglages du èsystme de écoordonnes 1(((G54)))
G55     utilise les érglages du èsystme de écoordonnes 2(((G55)))
G56     utilise les érglages du èsystme de écoordonnes 3(((G56)))
G57     utilise les érglages du èsystme de écoordonnes 4(((G57)))
G58     utilise les érglages du èsystme de écoordonnes 5(((G58)))
G59     utilise les érglages du èsystme de écoordonnes 6(((G59)))
G59.1   utilise les érglages du èsystme de écoordonnes 7(((G59.1)))
G59.2   utilise les érglages du èsystme de écoordonnes 8(((G59.2)))
G59.3   utilise les érglages du èsystme de écoordonnes 9(((G59.3)))
```

### 11.3.1 Système de coordonnées par défaut

Une autre variable dans le fichier VAR joue un rôle important dans les décalages, c'est la variable 5220. Dans les fichiers par défaut, sa valeur est fixée à 1,00000. Ce qui signifie que lorsque LinuxCNC démarre, il doit utiliser le premier système de coordonnées comme système par défaut. Si vous définissez celui-ci à 9,00000 le neuvième système décalé sera utilisé par défaut au démarrage du système et aux réinitialisations. Toute valeur autre qu'un entier compris entre 1 et 9, ou l'absence de la variable 5220, provoquera au démarrage le retour de LinuxCNC à la valeur par défaut de 1.00000.

### 11.3.2 Réglage des décalages avec G10

La commande G10 L2x peut être utilisée pour modifier les valeurs des décalages d'un système de coordonnées pièce: (Nous donnons seulement ici un bref aperçu, se reporter aux sections du G-code pour une description complète).

- *G10 L2 P(pièce 1-9)* - Ajuste les valeurs d'offset. La position courante reste inchangée. (voir la sectionG10 L2 pour les détails)
- *G10 L20 P(pièce 1-9)* - Ajuste les valeurs d'offset de sorte que la position courante devienne la position donnée en paramètre. (Voir la section G10 L20 pour les détails)

## 11.4 Décalages d'axes G92

G92 est la plus incomprise et la plus maligne des commandes programmables avec LinuxCNC. La façon dont elle fonctionne a un peu changé entre les premières versions et l'actuelle. Ces changements ont sans doute déconcerté de nombreux utilisateurs. Elle devrait être vue comme une commande produisant un décalage temporaire, qui s'applique à tous les autres décalages.

### 11.4.1 Les commandes G92

Ce jeu de commandes inclus:
– G92 - Cette commande, utilisée avec des mots d'axes, fixe les valeurs des variables de décalage.
– G92.1 - Cette commande met à zéro les valeurs des variables de G92.
– G92.2 - Cette commande suspend, sans les mettre à zéro, les variables de G92.
– G92.3 - Cette commande applique les valeurs de décalage qui ont été suspendues.
L'utilisateur doit bien comprendre le fonctionnement des valeurs de G92. Pour faire en sorte que le point actuel ait les coordonnées X0, Y0 et Z0 nous utiliserons *G92 X0 Y0 Z0*. G92 **ne fonctionne pas** depuis le système de coordonnées machine absolues. Il fonctionne à partir de **l'emplacement actuel**.

G92 travaille également à partir d'un emplacement actuel déjà modifié par tout autre décalage actif au moment où la commande G92 est invoquée. Lors de tests des différences entre les décalages de travail et les décalages réels, il a été constaté qu'un décalage G54 pouvait annuler un G92 et ainsi, donner l'apparence qu'aucun décalage n'était actif. Toutefois, le G92 était toujours actif, pour toutes les coordonnées et il a produit les décalages attendus pour tous les autres systèmes de coordonnées.

Lors du démarrage de LinuxCNC, si des offsets existent dans les variables de G92, ils seront appliqués lors de la prise d'origine des axes concernés. Il est donc de bonne pratique de mettre les offsets de G92 à zéro par G92.1 ou un G92.2 à la fin de leur utilisation.

### 11.4.2 Réglage des valeurs de G92

Il y a au moins deux façons d'établir les valeurs de G92.
– Par un clic droit de la souris sur les afficheurs de position de tklinuxcnc, une fenêtre s'ouvre dans laquelle il est possible de saisir une valeur.
– Par la commande G92.
Toutes les deux, fonctionnent depuis l'emplacement courant de l'axe auquel le déplacement doit être appliqué.

Programmer *G92 X Y Z A B C U V W* fixe les valeurs des variables de G92 de sorte que chaque axe prenne la valeur associée à son nom. Ces valeurs sont assignées à la position courante des axes. Ces résultats satisfont les paragraphes un et deux du document du NIST.

Les commandes G92 fonctionnent à partir de la position courante de l'axe, à laquelle elles ajoutent ou soustraient des valeurs pour donner à la position courante la valeur assignée par la commande G92. Elles prennent effet même si d'autres décalages sont déjà actifs.

Ainsi, si l'axe X est actuellement en position X=2.000, un *G92 X0* fixera un décalage de -2.0000, de sorte que l'emplacement actuel de X devienne X=0.000. Un nouveau *G92 X5.000* fixera un décalage de 3.000 et l'affichage indiquera une position courante X=5.000.

### 11.4.3 Précautions avec G92

Parfois, les valeurs de décalage d'un G92 restent bloquées dans le fichier VAR. Quand ça arrive, une ré-initialisation ou un redémarrage peut les rendre de nouveau actives. Les variables sont numérotées:

| Variable | Valeur |
|---|---|
| 5211 | 0.000000 |
| 5212 | 0.000000 |
| 5213 | 0.000000 |
| 5214 | 0.000000 |
| 5215 | 0.000000 |
| 5216 | 0.000000 |

où 5211 est le numéro du décalage de l'axe X et ainsi de suite. Si vous voyez des positions inattendues à la suite d'une commande de déplacement, ou même des chiffres inattendus dans l'affichage de la position lorsque vous démarrez, regardez ces variables dans le fichier VAR pour vérifier si elles contiennent des valeurs. Si c'est le cas, les mettre à zéro devrait solutionner le problème.

Si des valeurs G92 existent dans le fichier VAR quand LinuxCNC démarre, ces valeurs seront appliquées aux valeurs courantes

des emplacements d'axe. Si c'est sa position d'origine et que l'origine est définie au zéro machine, tout sera correct. Une fois que l'origine machine a été établie en utilisant les contacts d'origine machine, ou en déplaçant chaque axe à une position connue, puis en envoyant la commande de prise d'origine de l'axe, tous les décalages G92 seront appliqués. Si un X1 G92 est actif lors de la prise d'origine machine de l'axe X, la visu affichera *X: 1.000* au lieu du *X: 0.000* attendu, c'est parce-que le G92 a été appliqué à l'origine machine. Si vous passez un G92.1 et que la visu affiche tous à zéro, alors c'est que vous avez encore l'effet de l'offset G92 de la dernière session de LinuxCNC.

Sauf si votre intention est d'utiliser les mêmes décalages G92 dans le prochain programme, la meilleure pratique consiste à envoyer un G92.1 à la fin de tout fichier de G-code dans lequel vous utilisez les compensations G92.

## 11.5 Exemple de programme utilisant les décalages d'axes

Cet exemple de projet de gravure, usine un jeu de quatre cercles de rayon .1 pouce dans une forme grossière d'étoile au centre du cercle. Nous pouvons configurer individuellement les formes de la façon suivante:

```
G10 L2 P1 X0 Y0 Z0 (assure que G54 a mis la machine àé zro)
G0 X-0.1 Y0 Z0
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0
M2
```

Nous pouvons émettre une série de commandes pour créer des décalages pour les quatre autres cercles comme cela.

```
G10 L2 P2 X0.5 (édcalages G55 X la valeur de 0.5 pouces)
G10 L2 P3 X-0.5 (édcalages G56 X valeur de -0.5 pouces)
G10 L2 P4 Y0.5 (édcalages G57 valeur Y de 0.5 pouces)
G10 L2 P5 Y-0.5 (édcalages G58 valeur Y de -0.5 pouces)
```

Nous mettons ces ensembles dans le programme suivant:

```
(Un programme de fraisage de cinq petits cercles dans un losange)

G10 L2 P1 X0 Y0 Z0 (assure que G54 a mis la machine àé zro)
G10 L2 P2 X0.5 (édcalages G55 X la valeur de 0.5 pouces)
G10 L2 P3 X-0.5 (édcalages G56 X la valeur de -0.5 pouces)
G10 L2 P4 Y0.5 (édcalages G57 X la valeur de 0.5 pouces)
G10 L2 P5 Y-0.5 (édcalages G58 X la valeur de -0.5 pouces)

G54 G0 X-0.1 Y0 Z0 (cercle du centre)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G55 G0 X-0.1 Y0 Z0 (premier cercle écompens)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G56 G0 X-0.1 Y0 Z0 (èdeuxime cercle écompens)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G57 G0 X-0.1 Y0 Z0 (ètroisime cercle écompens)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G58 G0 X-0.1 Y0 Z0 (èquatrime cercle écompens)
```

```
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G54 G0 X0 Y0 Z0

M2
```

Maintenant c'est le moment d'appliquer une série de décalages G92 à ce programme. Vous verrez que c'est fait dans chaque cas de Z0. Si la machine était à la position zéro, un G92 Z1.0000 placé en tête de programme le décalerait d'un pouce. Vous pouvez également modifier l'ensemble du dessin dans le plan XY en ajoutant quelques décalages x et y avec G92. Si vous faites cela, vous devez ajouter une commande G92.1 juste avant le M2 qui termine le programme. Si vous ne le faites pas, les programmes que vous pourriez lancer après celui-ci, utiliseront également les décalages G92. En outre, cela permettrait d'éviter d'écrire les valeurs de G92 lorsque vous arrêtez LinuxCNC et donc, d'éviter de les recharger quand vous démarrez à nouveau le programme.

# Chapitre 12

# Les compensations d'outil

## 12.1 Compensation de longueur d'outil

### 12.1.1 Toucher

Dans la boîte de dialogue du bouton *Toucher* de l'interface AXIS, il est possible de mettre à jour automatiquement la table d'outils.

Séquence typique pour mise à jour de la table d'outils:
- Après la prise d'origine, charger un outil *Tn M6* dans lequel *n* est le numéro de l'outil.
- Déplacer l'outil pour établir le zéro pièce, en utilisant une cale d'épaisseur ou en faisant une petite passe puis une mesure.
- Cliquer sur le bouton *Toucher* de l'onglet *Controle manuel* (ou presser la touche *Fin* du clavier).
- Sélectionner *Table d'outils* dans la liste déroulante des systèmes de coordonnées.
- Entrer l'épaisseur de la cale ou la cote mesurée.
- Presser OK.

La table d'outil sera alors modifiée avec la longueur correcte en Z de l'outil. La visu affichera la position en Z correcte et une commande G43 sera passée pour que la nouvelle longueur Z de l'outil soit effective. Le choix *Table d'outils* n'apparaitra dans la liste déroulante du *Toucher*, que si l'outil à été chargé avec *Tn M6*.

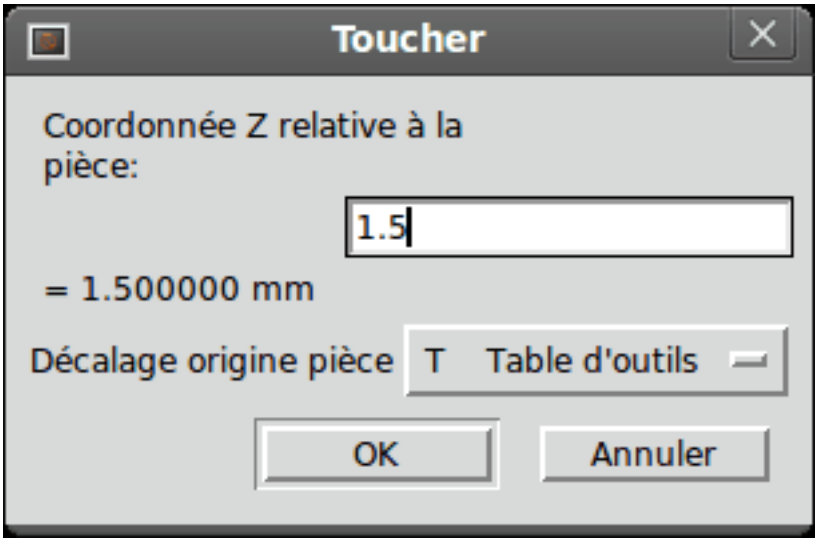



FIGURE 12.1 – Toucher et table d'outils

### 12.1.2 Utilisation de G10 L1/L10/L11

Les commandes G10 L1/L10/L11 peuvent être utilisées pour ajuster les compensations dans la table d'outils: (Ce sera juste une brève présentation, se reporter au guide de références du G-code pour des explications plus détaillées)

G10 L1 Pn - (n est le N° d'outil) Fixe les offsets de l'outil. La position courante n'est pas significative. Tous les détails ici.

G10 L10 Pn - (n est le N° d'outil) Fixe l'offset à la position courante, met les valeurs dans un système de 1 à 8. Tous les détails ici.

G10 L11 Pn - (n est le N° d'outil) Fixe l'offset à la position courante, met les valeurs dans le système 9. Tous les détails ici.

## 12.2 Table d'outils

La *table d'outils* est un fichier texte qui contient les informations de chaque outil. Ce fichier est placé dans le même répertoire que le fichier de configuration. Il est nommé *tool.tbl*. Les outils peuvent être dans un changeur d'outils ou changés manuellement. Le fichier peut être édité avec un éditeur de texte ou être mis à jour avec la commande G10 L1. Voir la section spécifique au tour pour ce qui concerne les outils de tour, avec un exemple de table. Le nombre d'outils est limité à 56 dans une table d'outils même si la numérotation des outils et des poches peut aller jusqu'à 99999.

### 12.2.1 Format de la table d'outils

TABLE 12.1: Format de la table d'outils

| **T#** | **P#** | **X** | **Y** | **Z** | **A** | **B** | **C** | **U** | **V** | **W** | **Dia** | **FA** | **BA** | **Ori** | **Rem** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ; | (aucune donnée après le point-virgule) | | | | | | | | | | | | | | |
| T1 | P17 | X0 | Y0 | Z0 | A0 | B0 | C0 | U0 | V0 | W0 | D0 | I0 | J0 | Q0 | ;rem |
| T2 | P5 | X0 | Y0 | Z0 | A0 | B0 | C0 | U0 | V0 | W0 | D0 | I0 | J0 | Q0 | ;rem |
| T3 | P12 | X0 | Y0 | Z0 | A0 | B0 | C0 | U0 | V0 | W0 | D0 | I0 | J0 | Q0 | ;rem |

En général, le format d'une ligne de table d'outils est le suivant:
- ; Un point-virgule comme premier caractère, aucune donnée
- T Numéro d'outil, 0-99999 (chaque numéro d'outil doit être unique)
- P Numéro de poche, 1-99999 (chaque numéro de poche doit être unique)
- X..W Offset d'outil sur les axes spécifiés - nombre à virgule flottante
- D diamètre d'outil - nombre à virgule flottante, valeur absolue
- I angle frontal (tour seulement) - nombre à virgule flottante
- J angle de dos (tour seulement) - nombre à virgule flottante
- Q orientation de l'outil (tour seulement) - entier de 0 à 9
- ; début de commentaire ou remarque - texte

Le fichier commence par un point-virgule en première ligne, suivi par les caractéristiques de 56 outils au maximum. [1]

Les versions antérieures de LinuxCNC avaient deux différents formats de table d'outils un pour les fraiseuses et un pour les tours, mais depuis la version 2.4.x, un format unique est utilisé pour toutes les machines. Il suffi d'ignorer les parties de la table d'outils qui ne se rapportent pas la machine actuelle, ou que vous n'avez pas besoin d'utiliser.

Chaque ligne du fichier de la table d'outils après le point-virgule ouvrant, contient les données pour un seul outil. Une ligne peut contenir jusqu'à 16 entrées, mais peut aussi en contenir beaucoup moins.

Les unités utilisées pour les longueurs, diamètres, etc. sont en unités machine.

1. Bien que les numéros d'outils puissent aller jusqu'à 99999, le nombre d'outils dans la table, en ce moment, est limité à un maximum de 56 outils pour des raisons techniques. Les développeurs de LinuxCNC envisagent la possibilité de faire sauter cette limitation. Si vous avez un très gros changeur d'outils, merci d'être patient.

Vous voudrez probablement maintenir les entrées d'outils dans l'ordre croissant, surtout si vous utilisez un changeur d'outils aléatoire. Bien que la table d'outils permettent des numéros d'outils dans n'importe quel ordre.

Chaque ligne peut avoir jusqu'à 16 valeurs. Les deux premières valeurs sont requises. La dernière valeur (un point-virgule suivi d'un commentaire) est optionnelle. La lecture sera rendue plus facile si les valeurs sont disposées en colonnes, comme indiqué dans le tableau, mais la seule exigence sur le format est qu'il y ait au moins un espace ou une tabulation après chacune des valeurs sur une ligne et un saut de ligne à la fin de chaque ligne.

La signification des valeurs et le type de données qu'elles contiennent sont les suivantes:

**Numéro d'outil (requis)** La colonne *T* contient un nombre entier non signé, qui représente le code de l'outil. L'opérateur peut utiliser n'importe quel code pour n'importe quel outil, tant que les codes sont des entiers non signés.

**Numéro de poche (requis)** La colonne *P* contient un nombre entier non signé, qui représente le numéro de poche (numéro de slot) du changeur d'outils, poche dans laquelle l'outil se trouve. Les entrées de cette colonne doivent être toutes différentes. Le numéro de poche commence typiquement à 1 et va au maximum de poches disponibles sur le changeur d'outils. Mais tous les changeurs d'outils ne suivent pas ce modèle. Votre numéro de poche sera déterminé, par le numéro que votre changeur d'outils utilisera pour se référer à ses poches. Tout cela pour dire que les numéros de poche que vous utiliserez seront déterminés par le schéma de numérotation de votre changeur d'outils. Les numéros de poche doivent suivre la même logique que la machine.

**Données d'offset des outils (optionnelles)** Les colonnes de données d'offset (XYZABCUVW) contiennent des nombres réels qui représentent les offsets d'outil pour chacun des axes. Ce nombre sera utilisé si, en usinage, les offsets de longueur d'outil sont utilisés et que l'outil concerné est sélectionné. Ces nombres peuvent être positif, égaux à zéro ou négatif, ils sont en fait, complètement optionnels. Bien qu'il vaudrait mieux qu'il y ait au moins une valeur ici, sinon il n'y aurait aucun intérêt à se servir d'une entrée complétement vide dans la table d'outils.

Sur une fraiseuse classique, on trouvera probablement une entrée en Z (offset de longueur d'outil). Sur un tour classique, on trouvera certainement un entrée en X (offset d'outil en X) et une en Z (offset d'outil en Z). Sur une fraiseuse classique utilisant la compensation de rayon d'outil, on trouvera une valeur en D pour l'offset de diamètre. Sur un tour classique utilisant la compensation de diamètre de bec d'outil, une valeur sera entrée en D (diamètre de bec).

Un tour demande encore d'autres information additionnelles pour décrire la forme et l'orientation de l'outil. Ainsi, sans tenir compte des angles ni des faces de l'outil, qui sont de la compétence du tourneur, on trouvera une valeur en I (angle avant) et en J (angle de dos) ainsi qu'une valeur en Q (orientation).

Une description complète des outils de tour ce trouve ici.

La colonne *Diamètre* contient un nombre réel. Ce nombre est utilisé seulement si la compensation est activée lors de l'usage de cet outil. Si la trajectoire programmée avec la compensation active, est un des bords de la matière à usiner, cette valeur doit être un nombre réel positif, représentant le diamètre mesuré de l'outil. Si la trajectoire programmée, toujours avec la compensation active, est prévue pour un diamètre nominal d'outil, ce nombre doit être très petit (négatif ou positif, mais proche de zéro), il représente seulement la différence entre le diamètre nominal et le diamètre mesuré de l'outil. Si la compensation n'est pas utilisée avec un outil, cette valeur est sans importance.

La colonne des commentaires peut optionnellement être utilisée pour décrire l'outil. Elle commence par un point-virgule, elle peut contenir n'importe quel texte pour le seul bénéfice de l'opérateur.

# Chapitre 13

# Fichier d'outils et compensations

## 13.1 Fichier d'outils

Les longueurs et diamètres d'outils peuvent être lus dans une table d'outils ou provenir d'un mot spécifié pour activer la compensation d'outil.

## 13.2 Compensation d'outil

La compensation d'outil peut causer beaucoup de problèmes aux meilleurs programmeurs. Mais elle peut aussi être une aide puissante quand elle est utilisée pour aider l'opérateur à obtenir une pièce à la cote. En réglant la longueur et le diamètre des outils dans une table d'outils, les décalages peuvent être utilisés pendant un cycle d'usinage qui tient compte des variations de taille de l'outil, ou pour des déviations mineures des trajectoires programmées. Et ces changements peuvent être faits sans que l'opérateur n'ait à changer quoi que ce soit dans le programme.

Tout au long de ce module, vous trouverez occasionnellement des références à des fonctions canoniques, là où il est nécessaire pour le lecteur de comprendre comment fonctionne une compensation d'outil dans une situation spécifique. Ces références ont pour but de donner au lecteur une idée de la séquences plutôt que d'exiger qu'il comprenne la façon dont les fonctions canoniques elles-mêmes fonctionnent dans LinuxCNC.

## 13.3 Compensation de longueur d'outil

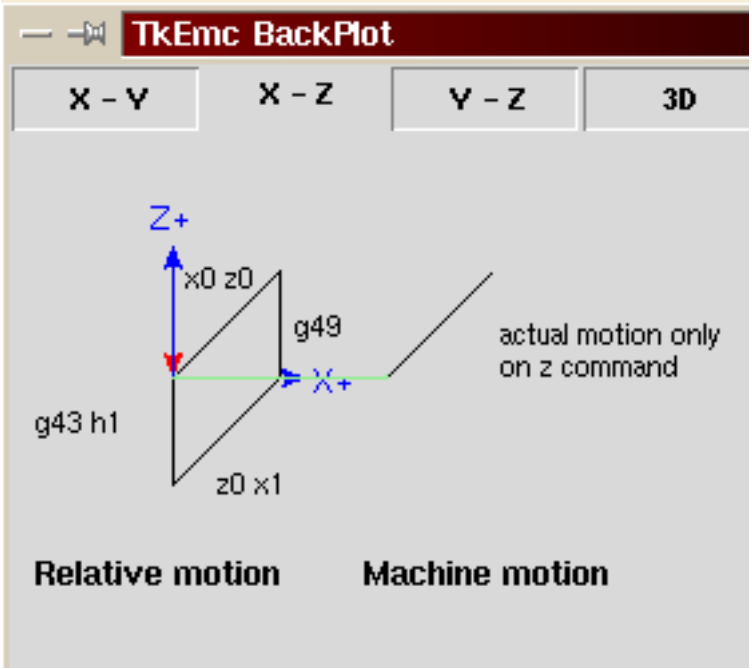
Les compensations de longueur d'outil sont données comme des nombres positifs dans la table d'outils. Une compensation d'outil est programmée en utilisant G43 Hn, où n est le numéro d'index de l'outil souhaité dans la table d'outil. Il est prévu que toutes les entrées dans la table d'outils soit positives. La valeur de H est vérifiée, elle doit être un entier non négatif quand elle est lue. L'interpréteur se comporte comme suit:

1. Si G43 Hn est programmé, un appel à la fonction USE_TOOL_LENGTH_OFFSET(longueur) est fait (où longueur est l'écart de longueur, lu dans la table d'outils, de l'outil indexé n), tool_length_offset est repositionné dans le modèle de réglages de la machine et la valeur de current_z dans le modèle est ajustée. Noter que n n'a pas besoin d'être le même que le numéro de slot de l'outil actuellement dans la broche.
2. Si G49 est programmé, USE_TOOL_LENGTH_OFFSET(0.0) est appelée, tool_length_offset est remis à 0.0 dans le modèle de réglages de la machine et la valeur courante de current_z dans le modèle est ajustée. L'effet de la compensation de longueur d'outil est illustrée dans la capture ci-dessous. Noter que la longueur de l'outil est soustraite de Z pour que le point contrôlé programmé corresponde à la pointe de l'outil. Il faut également noter que l'effet de la compensation de longueur est immédiat quand on voit la position de Z comme une coordonnée relative mais il est sans effet sur la position actuelle de la machine jusqu'à ce qu'un mouvement en Z soit programmé.

Programme de test de longueur d'outil.

L'outil #1 fait un pouce de long.

```
N01 G1 F15 X0 Y0 Z0
N02 G43 H1 Z0 X1
N03 G49 X0 Z0
N04 G0 X2
N05 G1 G43 H1 G4 P10 Z0 X3
N06 G49 X2 Z0
N07 G0 X0
```




Avec ce programme, dans la plupart des cas, la machine va appliquer le décalage sous forme d'une rampe pendant le mouvement en xyz suivant le mot G43.

## 13.4 Compensation de rayon d'outil

La compensation de rayon d'outil permet de suivre un parcours sans se préoccuper du diamètre de l'outil. La seule restriction, c'est que les *mouvements d'entrée* doivent être au moins aussi long que le rayon de l'outil utilisé.

Il y a deux parcours que l'outil peut prendre pour usiner un profil quand la compensation de rayon est activée, un parcours à gauche du profil et un à droite du profil. Pour les visualiser, il faut s'imaginer être debout sur la pièce, marchant en suivant l'outil pendant que celui-ci progresse dans la matière. G41 fait passer l'outil à gauche du profil et G42 le fait passer à droite du profil.

Le point final de chaque mouvement, dépends du mouvement suivant. Si le mouvement suivant crée un angle extérieur, le mouvement se terminera à l'extrémité de la ligne de coupe compensée. Si le mouvement suivant crée un angle intérieur, l'outil s'arrêtera avant d'interférer avec la matière de la pièce. La figure suivante montre comment le mouvement se termine à différents endroits, dépendants du mouvement suivant.

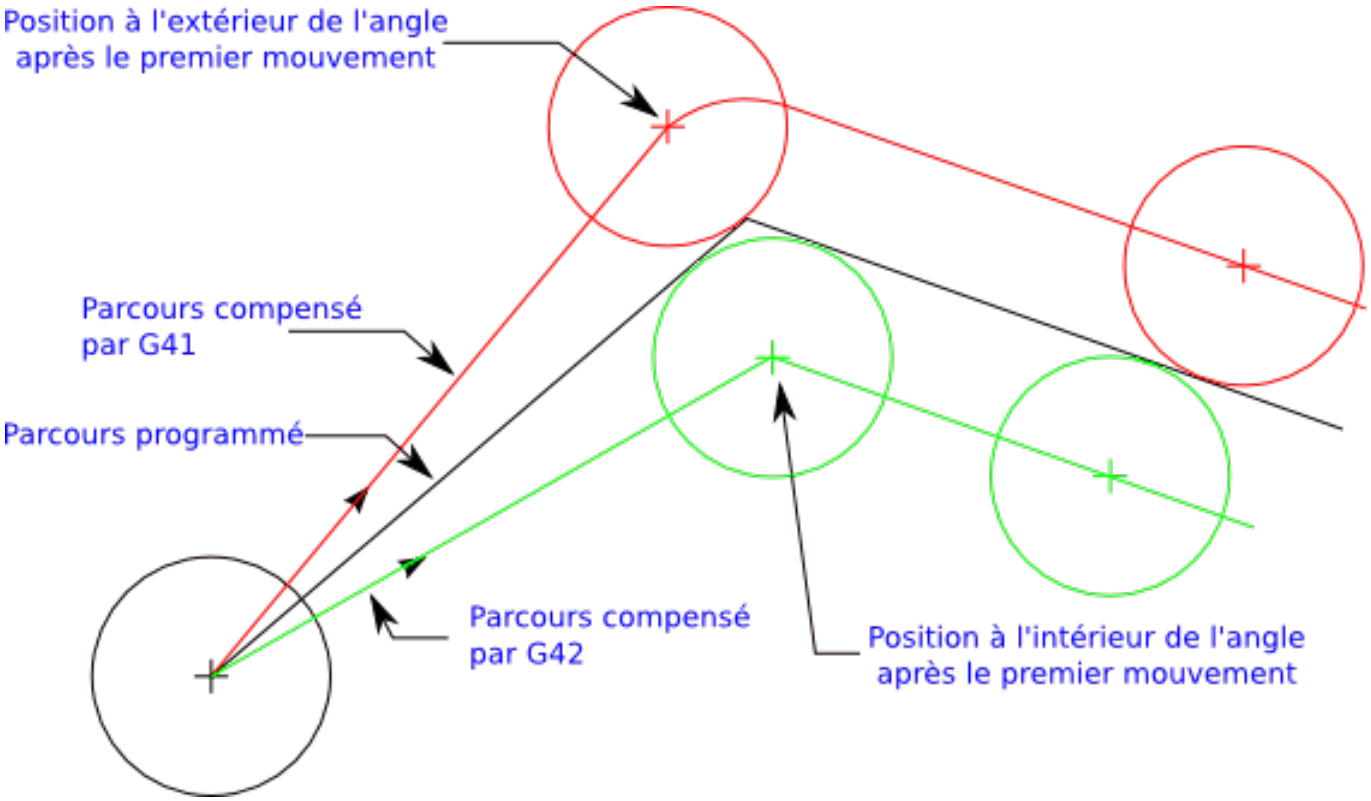



FIGURE 13.1 – Point final de la compensation

### 13.4.1 Vue générale

#### 13.4.1.1 Table d'outils

La compensation de rayon d'outil utilise les données de la table d'outils pour déterminer le décalage nécessaire. Les données peuvent être introduites à la volée, avec G10 L1.

#### 13.4.1.2 Programmation des mouvements d'entrée

Tout mouvement suffisamment long pour arriver en position compensée, sera un mouvement d'entrée valide. La longueur minimale équivaut au rayon de l'outil. Ça peut être un mouvement en vitesse rapide au dessus de la pièce. Si plusieurs mouvements en vitesse rapide sont prévus après un G41/G42, seul le dernier placera l'outil en position compensée.

Dans la figure suivante, on voit que le mouvement d'entrée est compensé à droite du profil. Ce qui aura pour effet, lors du mouvement d'entrée, de déplacer le centre de l'outil, d'un rayon d'outil à droite de X0. Dans ce cas, le mouvement d'entrée laissera un petit plot de matière en raison du décalage de compensation et de l'arrondi de l'outil.

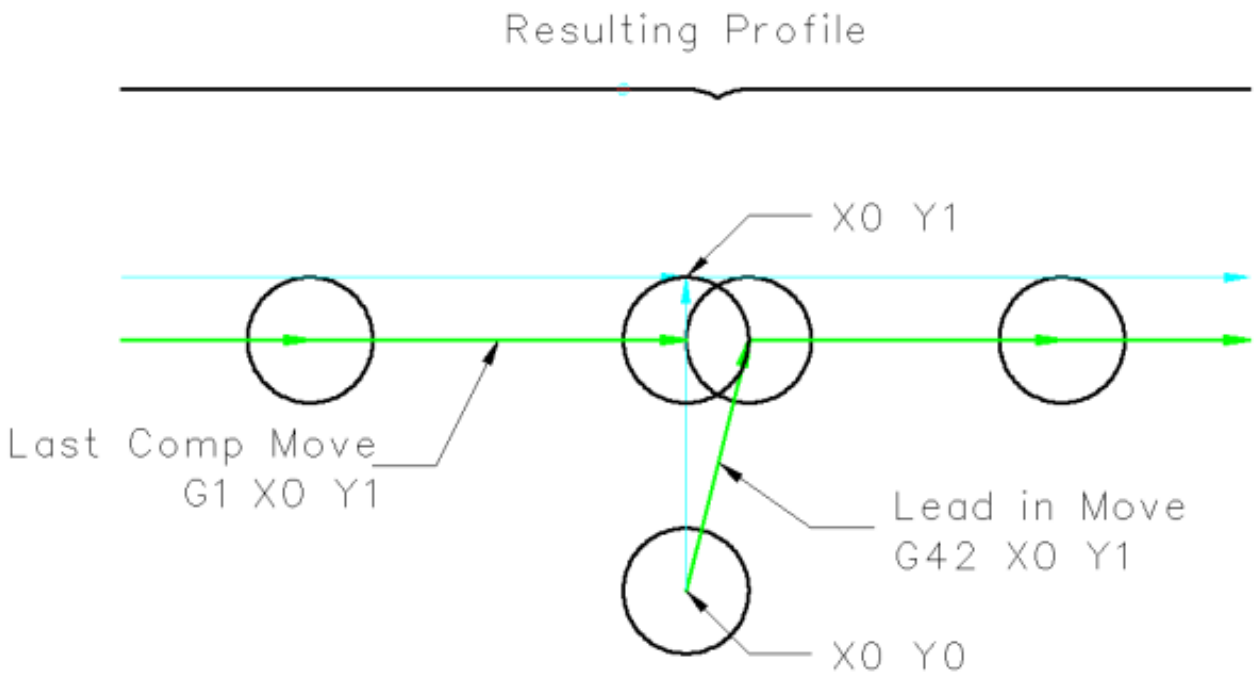



FIGURE 13.2 – Mouvement d'entrée

#### 13.4.1.3 Mouvement en Z

Un mouvement en Z est possible pendant que le contour est suivi dans le plan XY. Des portions du contour peuvent être sautées en rétractant l'axe Z au dessus du bloc et en amenant Z au dessus du prochain point de départ.

#### 13.4.1.4 Mouvement en vitesse rapide

Des mouvements en vitesse rapide peuvent être programmé avec les compensations d'outil actives.

#### 13.4.1.5 Bonne pratique

– Débuter tout programme avec un G40 pour être sûr que la compensation est désactivée.

### 13.4.2 Exemples de profils

#### 13.4.2.1 Profil extérieur

```
G-Code
F25  (Vitesse d'avance travail)
G40  (Révocation des compensations)
G10 L1 P1 R.25 Z1  (Réglages en table d'outils)
T1 M6  (Appel de l'outil 1)
G42  (Compensation d'outil à droite du profil)
G1 X1 Y1  (Mouvement d'entrée)
X3  (Parcours trajectoire de coupe)
Y3
X1
Y1
G40  (Révocation de la compensation)
G0 Y0 Y0  (Dégagement de l'outil)
M2  (Fin de programme)
```

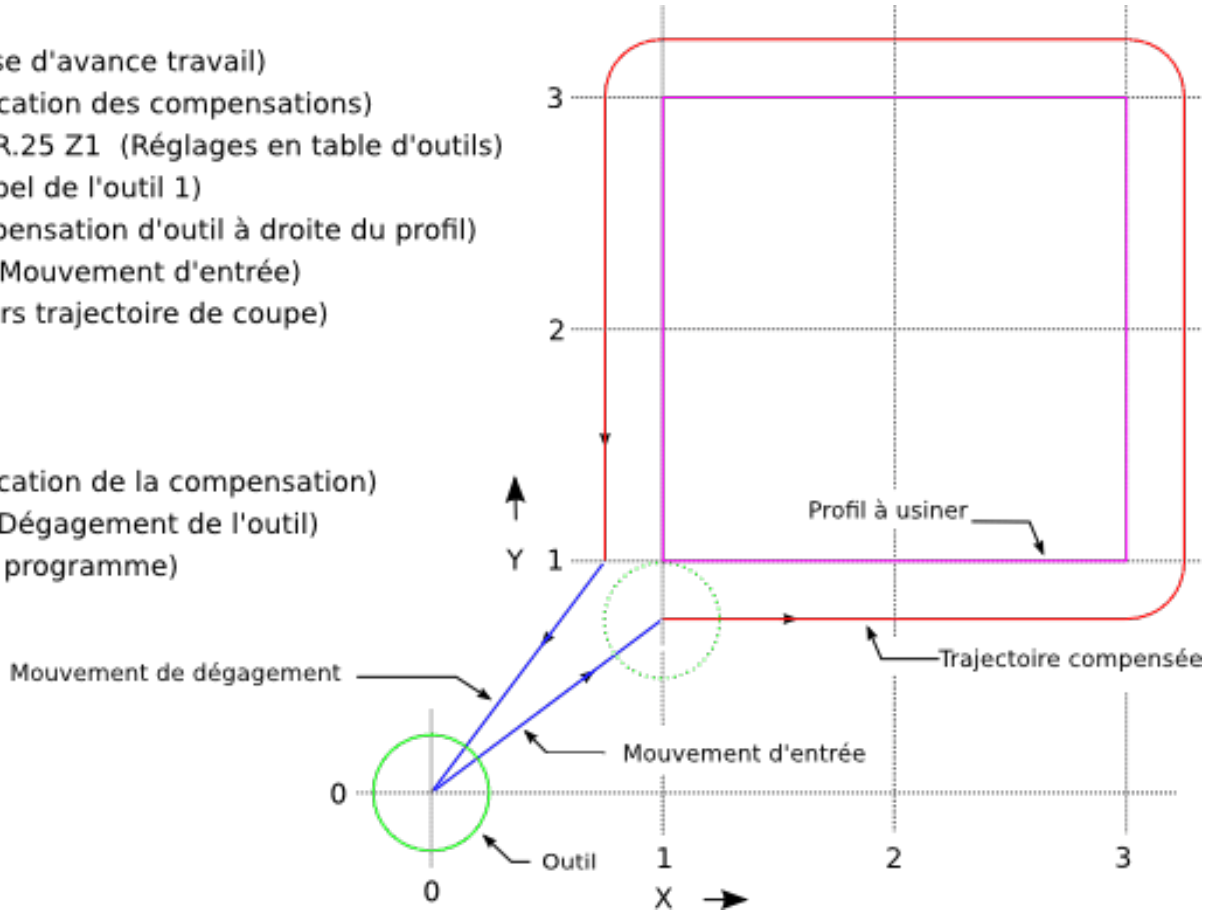



FIGURE 13.3 – Profil extérieur

#### 13.4.2.2 Profil intérieur

```
G-Code
F30  (Sélection de la vitesse)
G10 L1 P1 R.25 Z1  (Réglages en table d'outils)
T1 M6  (Chargement de l'outil 1)
G0 Z0  (Dégagement de l'outil sur l'axe Z)
G41  (Compensation d'outil à gauche du profil)
X3 Y2  (Déplacement rapide sur point de départ)
G1 X4 Z-1  (Plongée vers la profondeur de coupe)
G3 X5 Y3 J1  (Arc d'entrée)
G1 Y5  (Parcours sur la trajectoire de coupe)
X1
Y1
X5
Y3
G3 X4 Y4 I-1  (Arc de sortie)
G0 Z0  (Dégagement de l'outil sur axe Z)
G40  (Désactivation de la compensation d'outil)
G0 X1 Y1  (Déplacement sur position sécurisée)
T0 M6  (Déchargement de l'outil)
M2  (Fin de programme)
```

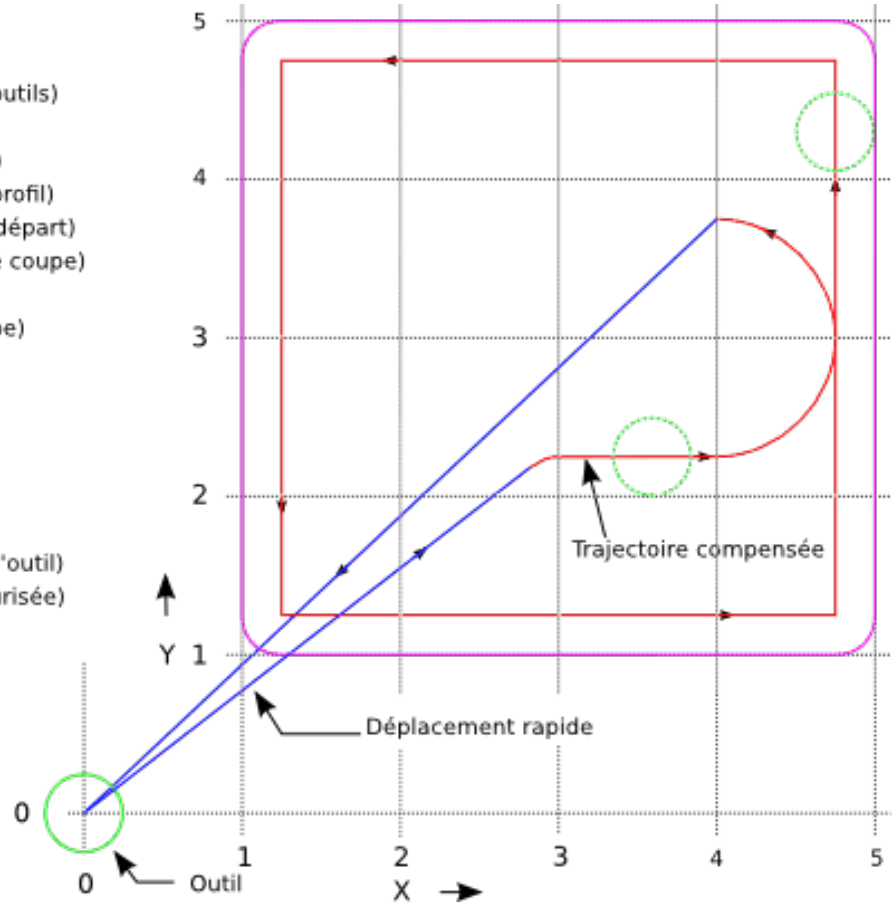



FIGURE 13.4 – Profil intérieur

## 13.5 Deux exposés sur les compensations d'outil

Ces deux exposés ont été écrits par des experts de la CNC, nous pensons que leur lecture sera très utile.

By Jon Elson

La compensation de rayon d'outil (également appelée compensation de diamètre d'outil) à été ajoutée aux spécifications RS-274D à la demande d'utilisateurs, car elle est extrêmement utile, mais son implémentation a été assez mal pensée. L'objectif de cette fonctionnalité est de permettre aux programmeurs de *virtualiser* la trajectoire de l'outil, de sorte que la machine puisse pendant toute l'exécution, déterminer le bon décalage a apporter à la position de l'outil pour respecter les cotes, en s'appuyant sur les diamètres d'outils existants. Si un outil est réaffuté, son diamètre sera légèrement plus petit que celui d'origine, il faudra également en tenir compte.

Le problème est pour donner à la machine la trajectoire exacte où l'outil doit usiner, sur le côté intérieur d'un parcours imaginaire, ou sur le côté extérieur. Comme ces trajectoires ne sont pas nécessairement fermées (même si elles peuvent l'être), il est quasiment impossible à la machine de connaître de quel côté du profil elle doit compenser l'outil. Il a été décidé qu'il n'y aurait que deux choix possibles, outil à *gauche* du profil à usiner et outil à *droite* du profil à usiner. Ce qui doit être interprété à gauche ou à droite du profil à usiner en suivant l'outil le long du profil.

### 13.5.1 Compensation de rayon d'outil, détails

By Tom Kramer and Fred Proctor

Les possibilités de compensation de rayon d'outil de l'interpréteur, autorise le programmeur à spécifier si l'outil doit passer à gauche ou à droite du profil à usiner. Ce profil peut être un contour ouvert ou fermé, dans le plan XY composé de segments en arcs de cercles et en lignes droites. Le contour peut être le pourtour d'une pièce brute ou, il peut être une trajectoire exacte suivie par un outil mesuré avec précision. La figure ci-dessous, montre deux exemples de trajectoires d'usinage d'une pièce triangulaire, utilisant la compensation de rayon d'outil.

Dans les deux exemples, le triangle gris représente le matériau restant après usinage et la ligne extérieure représente le parcours suivi par le centre de l'outil. Dans les deux cas le triangle gris est conservé. Le parcours de gauche (avec les coins arrondis) est le parcours généralement interprété. Dans la méthode de droite (celle marquée Not this way), l'outil ne reste pas en contact avec les angles vifs du triangle gris.

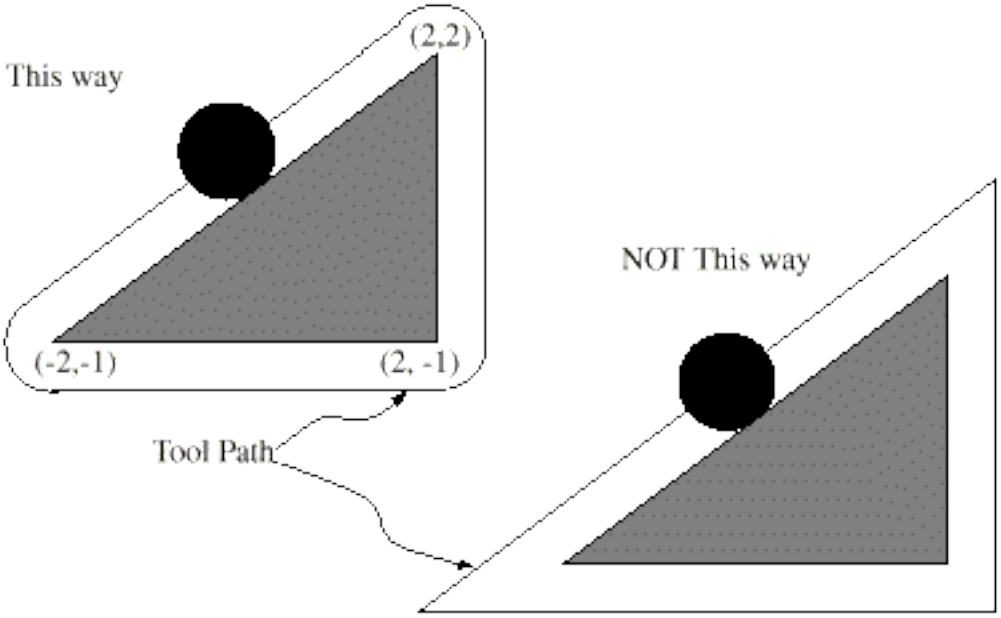



FIGURE 13.5 –

Des mouvements sur l'axe Z peuvent avoir lieu pendant que le contour est suivi dans le plan XY. Des portions du contour peuvent être franchies avec l'axe Z en retrait au dessus de la pièce pendant la poursuite du parcours et jusqu'au point où l'usinage doit reprendre, l'axe Z plongera de nouveau en position. Ces dégagements de zones non usinées peuvent être faits en vitesse travail (G1), en rapide (G0), en vitesse inverse du temps (G93) ou en avance en unités par minute (G94) toutes peuvent être utilisées avec la compensation de rayon d'outil. Sous G94, la vitesse sera appliquée à la pointe de l'outil coupant, non au contour programmé.

#### 13.5.1.1 Instructions de programmation

- Pour activer la compensation de rayon d'outil, programmer soit, G41 (pour maintenir l'outil à gauche du profil à usiner) soit, G42 (pour maintenir l'outil à droite du profil). Dans la figure figure:7 précédente, par exemple, si G41 était programmé, l'outil devrait tourner en sens horaire autour du triangle et dans le sens contraire si G42 était programmé.
- Pour désactiver la compensation de rayon d'outil, programmer G40.
- Si un G40, G41, ou G42 est programmé dans la même ligne qu'un mouvement d'axe, la compensation de rayon sera activée ou désactivée avant que le mouvement ne soit fait. Pour que le mouvement s'effectue en premier, il doit être programmé séparément et avant.

#### 13.5.1.2 La valeur de D

L'interpréteur actuel requiert une valeur D sur chaque ligne contenant un mot G41 ou G42. Le nombre D doit être un entier positif. Il représente le numéro de slot de l'outil, dont le rayon (la moitié du diamètre d'outil indiqué dans la table d'outils) sera

compensé. Il peut aussi être égal à zéro, dans ce cas, la valeur du rayon sera aussi égale à zéro. Toutes les poches de la table d'outils peuvent être sélectionnés de cette façon. Le nombre D n'est pas nécessairement le même que le numéro de poche de l'outil monté dans la broche.

#### 13.5.1.3 Table d'outils

La compensation de rayon d'outil utilise les données fournies par la table d'outils de la machine. Pour chaque poche d'outil dans le carrousel, la table d'outils contient le diamètre de l'outil rangé à cet emplacement (ou la différence entre le diamètre nominal de l'outil placé dans cette poche et son diamètre actuel). La table d'outils est indexée par les numéros de poche. Reportez vous à la page des *Fichiers d'outils* pour savoir comment remplir ces fichiers.

#### 13.5.1.4 Deux types de contour

L'interpréteur contrôle la compensation pour deux types de contour:
- 1) Le contour donné dans le code NC correspond au bord extérieur du matériau après usinage. Nous l'appellerons *contour sur le profil du matériau*.
- 2) Le contour donné dans le code NC correspond à la trajectoire suivie par le centre d'un outil de rayon nominal. Nous l'appellerons *contour sur le parcours d'outil*.

L'interpréteur ne dispose d'aucun paramètre pour déterminer quel type de contour est utilisé, mais la description des contours est différente (pour la même géométrie de pièce) entre les deux types, les valeurs des diamètres dans la table d'outils seront également différentes pour les deux types.

#### 13.5.1.5 Contour sur le profil du matériau

Lorsque le contour est sur le profil du matériau, c'est ce tracé qui est décrit dans le programme NC. Pour un contour sur le profil du matériau, la valeur du diamètre dans la table d'outils correspond au diamètre réel de l'outil courant. Cette valeur dans la table doit être positive. Le code NC pour ce type de contour reste toujours le même à l'exception du diamètre de l'outil (actuel ou nominal).

Exemple 1 :

Voici un programme NC qui usine le matériau en suivant le profil d'un des triangles de la figure précédente. Dans cet exemple, la compensation de rayon est celle du rayon actuel de l'outil, soit 0.5". La valeur pour le diamètre dans la table d'outil est de 1.0".

```
N0010 G41 G1 X2 Y2 (active la compensation et fait le mouvement d'éentre)
N0020 Y-1 (suit la face droite du triangle)
N0030 X-2 (suit la base du triangle)
N0040 X2 Y2 (suit l'éhypotnuse du triangle)
N0050 G40 (édsactive la compensation)
```

Avec ce programme, l'outil suit cette trajectoire: un mouvement d'entrée, puis la trajectoire montrée dans la partie gauche de la figure, avec un déplacement de l'outil en sens horaire autour du triangle. Noter que les coordonnées du triangle de matériau apparaissent dans le code NC. Noter aussi que la trajectoire inclus trois arcs qui ne sont pas explicitement programmés, ils sont générés automatiquement.

#### 13.5.1.6 Contour sur le parcours d'outil

Lorsque le contour est sur le parcours d'outil, la trajectoire décrite dans le programme correspond au parcours que devra suivre le centre de l'outil. Le bord de l'outil, à un rayon de là, (excepté durant les mouvements d'entrée) suivra la géométrie de la pièce. La trajectoire peut être créée manuellement ou par un post-processeur, selon la pièce qui doit être réalisée. Pour l'interpréteur, le parcours d'outil doit être tel que le bord de l'outil reste en contact avec la géométrie de la pièce, comme montré à gauche de la figure 7. Si une trajectoire du genre de celle montrée sur la droite de la figure 7 est utilisée, dans laquelle l'outil ne reste pas en permanence au contact avec la géométrie de la pièce, l'interpréteur ne pourra pas compenser correctement si un outil en dessous de son diamètre nominal est utilisé.

Pour un contour sur le parcours d'outil, la valeur pour le diamètre de l'outil dans la table d'outils devra être un petit nombre positif si l'outil sélectionné est légèrement sur-dimensionné. La valeur du diamètre sera un petit nombre négatif si l'outil est

légèrement sous-dimensionné. Si un diamètre d'outil est négatif, l'interpréteur compense de l'autre côté du contour programmé et utilise la valeur absolue donnée au diamètre. Si l'outil courant est à son diamètre nominal, la valeur dans la table d'outil doit être à zéro.

Exemple de contour sur le parcours d'outil

Supposons que le diamètre de l'outil courant monté dans la broche est de 0.97 et le diamètre utilisé en générant le parcours d'outil a été de 1.0 . Alors la valeur de diamètre dans la table d'outils pour cet outil est de -0.03. Voici un programme G-code qui va usiner l'extérieur d'un triangle de la figure 7.

```
N0010 G1 X1 Y4.5 (mouvement d'alignement)
N0020 G41 G1 Y3.5 (active la compensation et premier mouvement d'éentre)
N0030 G3 X2 Y2.5 I1 (èdeuxime mouvement d'éentre)
N0040 G2 X2.5 Y2 J-0.5 (usinage le long de l'arc en haut du parcours d'outil)
N0050 G1 Y-1 (usinage le long du ôéct droit du parcours d'outil)
N0060 G2 X2 Y-1.5 I-0.5 (usinage de l'arc en bas à droite)
N0070 G1 X-2 (usinage de la base du parcours d'outil)
N0080 G2 X-2.3 Y-0.6 J0.5 (usinage de l'arc en bas à gauche)
N0090 G1 X1.7 Y2.4 (usinage de l'éhypotnuse)
N0100 G2 X2 Y2.5 I0.3 J-0.4 (usinage de l'arc en haut à droite)
N0110 G40 (édsactive la compensation)
```

Avec ce programme, l'outil suit cette trajectoire: un mouvement d'alignement, deux mouvements d'entrée, puis il suit une trajectoire légèrement intérieure au parcours d'outil montré sur la figure 7 en tournant en sens horaire autour de la pièce. Cette compensation est à droite de la trajectoire programmée, même si c'est G41 qui est programmé, en raison de la valeur négative du diamètre.

#### 13.5.1.7 Erreurs de programmation et limitations

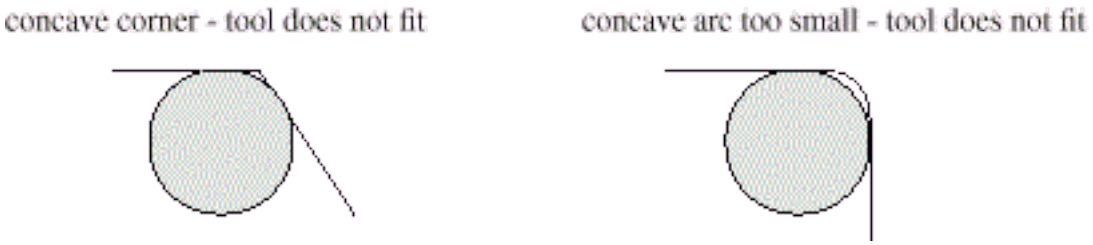
Les messages en rapport avec la compensation de rayon d'outil, délivrés par l'interpréteur sont les suivants:
- Impossible de changer les décalages d'axes avec la compensation de rayon d'outil
- Impossible de changer d'unité avec la compensation de rayon d'outil
- Impossible d'activer la compensation de rayon d'outil en dehors du plan XY
- Action impossible, la compensation de rayon d'outil est déjà active
- Impossible d'utiliser G28 ou G30 avec la compensation de rayon d'outil
- Impossible d'utiliser G53 avec la compensation de rayon d'outil
- Impossible d'utiliser le plan XZ avec la compensation de rayon d'outil
- Impossible d'utiliser le plan YZ avec la compensation de rayon d'outil
- Coin concave avec la compensation de rayon d'outil
- Interférence de l'outil avec une partie finie de la pièce avec la compensation de rayon d'outil [1]
- Mot D sur une ligne sans mot de commande G41 ni G42
- Index de rayon d'outil trop grand
- Le rayon de l'outil n'est pas inférieur au rayon de l'arc avec la compensation de rayon
- Deux G-codes du même groupe modal sont utilisés.

Pour certains de ces messages, des explications sont données plus loin.

Changer d'outil alors que la compensation de rayon d'outil est active n'est pas considéré comme une erreur, mais il est peu probable que cela soit fait intentionnellement. Le rayon d'outil utilisé lors de l'établissement de la compensation continuera à être utilisé jusqu'à la désactivation de la compensation, même si un nouvel outil est effectivement utilisé.

Quand la compensation de rayon d'outil est active, il est physiquement possible de faire un cercle, dont le rayon est la moitié du diamètre de l'outil donné dans la table d'outils, il sera tangent avec l'outil en tout point de son contour.

1. Le terme anglais *gouging* indique une interférence entre l'outil et une partie finie de la pièce ou la paroi d'une cavité. Par extension, le terme est parfois repris pour une interférence entre le porte-outil ou la broche et la pièce.




En particulier, l'interpréteur traite les coins concaves et les arcs concaves plus petits que l'outil, comme des erreurs, le cercle ne peut pas être maintenu tangent avec le contour dans ces situations. Cette détection de défaut, ne limite pas les formes qui peuvent être usinées, mais elle requiert que le programmeur précise la forme exacte à usiner (ou le parcours d'outil qui doit être suivi) et non une approximation. A cet égard, l'interpréteur utilisé par LinuxCNC diffère des interpréteurs utilisés dans beaucoup d'autres contrôleurs, qui passent ces erreurs sous silence et laissent l'outil interférer avec la partie finie de la pièce (gouging) ou arrondissent des angles qui devraient être vifs. Il n'est pas nécessaire, de déplacer l'outil entre la désactivation de la compensation et sa réactivation, mais le premier mouvement suivant la réactivation sera considéré comme premier mouvement, comme déjà décrit plus tôt.

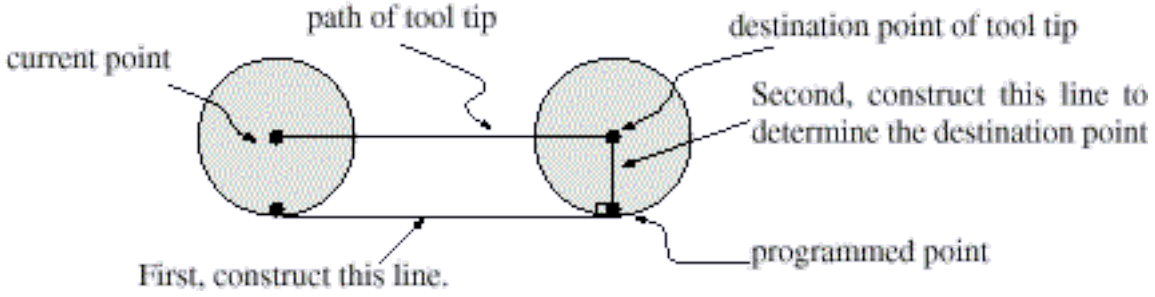
Il n'est pas possible de passer d'un index de rayon d'outil à un autre alors que la compensation est active. Il est également impossible de basculer la compensation d'un côté à l'autre avec la compensation active. Si le prochain point de destination XY est déjà dans le périmètre d'action de l'outil quand la compensation est activée, le message indiquant une interférence outil/surface finie, s'affichera quand la ligne du programme qui donne cette destination sera atteinte. Dans ce cas, l'outil a déjà usiné dans le matériau, là où il n'aurait pas dû…

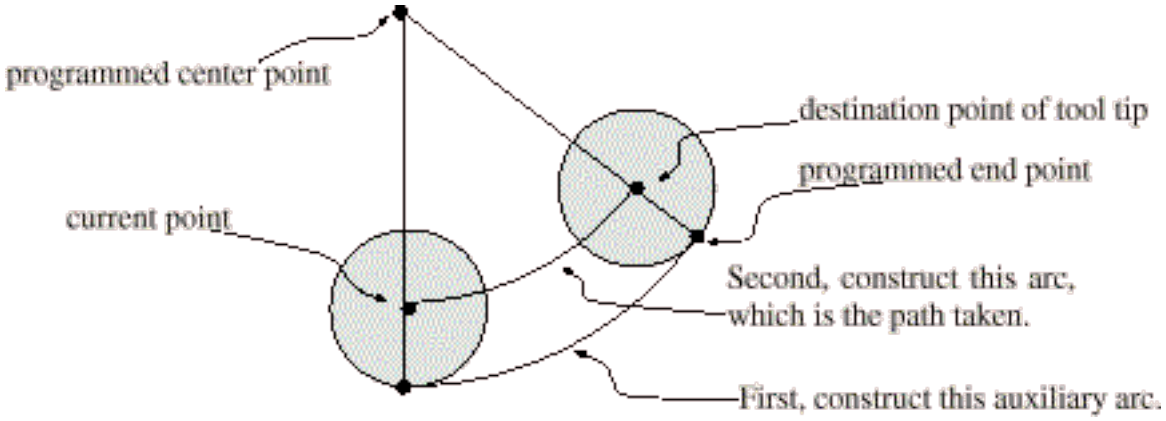
Si le numéro de slot programmé par le mot D est supérieur au nombre d'emplacements disponibles dans le carrousel, un message d'erreur sera affiché. Dans l'implémentation actuelle, le nombre d'emplacements maximum est de 68.

Le message d'erreur *Deux G-codes du même groupe modal sont utilisés* est un message générique utilisé pour plusieurs jeux de G-codes. Il s'applique à la compensation de rayon d'outil, il signifie que plus d'un code G40, G41 ou G42 apparaît sur la même ligne de programme NC, ce qui n'est pas permis.

### 13.5.2 Premier mouvement

L'algorithme utilisé lors du premier déplacement, quand c'est une ligne droite, consiste à tracer une droite, depuis le point d'arrivée, tangente à un cercle dont le centre est le point actuel, et le rayon, celui de l'outil. Le point de destination de la pointe de l'outil se trouve alors au centre d'un cercle de même rayon, tangent à la ligne droite tracée précédemment. C'est montré sur la figure 9. Si le point programmé est situé à l'intérieur de la première section d'outil (le cercle de gauche), une erreur sera signalée.






Si le premier mouvement après que la compensation de rayon d'outil a été activée est un arc, l'arc qui sera généré est dérivé d'un arc auxiliaire, qui a son centre identique à celui du point central programmé, passe par le point final de l'arc programmé et, est tangent à l'outil à son emplacement courant. Si l'arc auxiliaire ne peut pas être construit, une erreur sera signalée. L'arc généré déplacera l'outil pour qu'il reste tangent à l'arc auxiliaire pendant tout le mouvement. C'est ce que montre sur la figure 10.

Indépendamment du fait que le premier déplacement est une droite ou un arc, l'axe Z peut aussi se déplacer en même temps. Il se déplacera linéairement, comme c'est le cas quand la compensation de rayon n'est pas utilisée. Les mouvements des axes rotatifs (A, B et C) sont autorisés avec la compensation de rayon d'outil, mais leur utilisation serait vraiment très inhabituelle.

Après les mouvements d'entrée en compensation de rayon d'outil, l'interpréteur maintiendra l'outil tangent au contour programmé et du côté approprié. Si un angle aigu se trouve dans le parcours, un arc est inséré pour tourner autour de l'angle. Le rayon de cet arc sera de la moitié du diamètre de l'outil donné dans la table d'outils.

Quand la compensation de rayon est désactivée, aucun mouvement de sortie particulier n'est fait. Le mouvement suivant sera ce qu'il aurait été si la compensation n'avait jamais été activée et que le mouvement précédent ait placé l'outil à sa position actuelle.

#### 13.5.2.1 Programmation des mouvements d'entrée

En général, un mouvement d'alignement et deux mouvements d'entrée sont demandés pour commencer la compensation correctement. Cependant, si le contour programmé comporte des pointes et des angles aigus, un seul mouvement d'entrée (plus, éventuellement, un mouvement de pré-entrée) est demandé. La méthode générale, qui fonctionne dans toutes les situations, est décrite en premier. Elle suppose que le programmeur connait déjà le contour et son but est d'ajouter le mouvement d'entrée.

#### 13.5.2.2 Méthode générale

La méthode générale de programmation comprend un mouvement d'alignement et deux mouvements d'entrée. Les mouvements d'entrée expliqués ci-dessus, seront repris comme exemple. Voici le code correspondant:

```
N0010 G1 X1 Y4.5 (mouvement d'alignement vers le point C)
N0020 G41 G1 Y3.5 (active la compensation et fait le premier mouvement
d'éentre vers le point B)
N0030 G3 X2 Y2.5 I1 (fait le second mouvement d'éentre vers le point A)
```

Voir la figure 11. La figure montre les deux mouvements d'entrée mais pas le mouvement d'alignement.

En premier, choisir un point A sur le contour où il convient d'attacher un arc d'entrée. Spécifier un arc à l'extérieur du contour qui commence au point B et s'achève au point A, tangent au contour (et aller dans la même direction que celle prévue pour tourner autour du contour). Le rayon doit être supérieur à la moitié du diamètre donné dans la table d'outils. Ensuite, tirer une ligne tangente à l'arc, du point B au point C, placé de telle sorte que la ligne BC fasse plus d'un rayon de long.

Après que la construction soit terminée, le code est écrit dans l'ordre inverse de celui de la construction. La compensation de rayon d'outil est activée après le mouvement d'alignement et avant le premier mouvement d'entrée. Dans le code précédent, la ligne N0010 fait le mouvement d'alignement, la ligne N0020 active la compensation et fait le premier mouvement d'entrée et la ligne N0030 fait le second mouvement d'entrée.

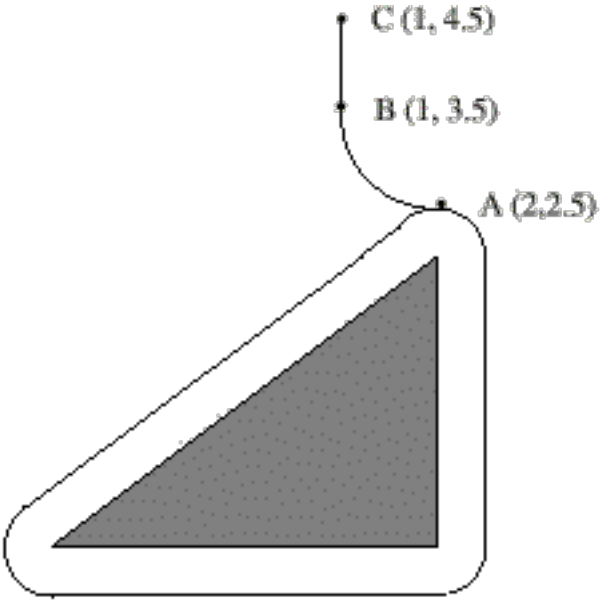



**Figure 11. Cutter Radius Compensation Entry Moves**

Dans cet exemple, l'arc AB et la ligne BC sont très larges, ce n'est pas nécessaire. Pour un contour sur parcours d'outil, le rayon de l'arc AB demande juste à être légèrement plus grand que la variation maximale du rayon de l'outil par rapport à son rayon nominal. Également, pour un contour sur parcours d'outil, le côté choisi pour la compensation doit être celui utilisé si l'outil est sur-dimensionné. Comme mentionné précédemment, si l'outil est sous-dimensionné, l'interpréteur basculera de l'autre côté.

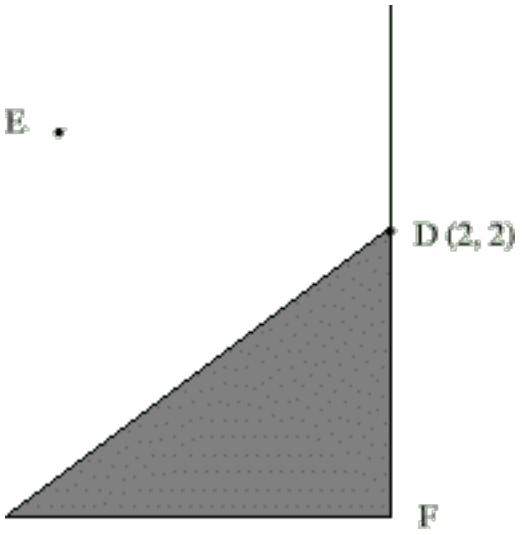
#### 13.5.2.3 Méthode simple

Si le contour est sur le profil du matériau et qu'il comprends des angles aigus quelque part sur le contour, une méthode simple pour faire l'entrée est possible. Voir la figure 12.

Premièrement, choisir un angle aigu, par exemple D. Ensuite, décider comment on va tourner autour du matériau depuis le point D. Dans notre exemple nous maintiendrons l'outil à gauche du profil et nous avancerons vers F. Prolonger la ligne FD (si le segment suivant du contour est un arc, prolonger la tangente à l'arc FD depuis D) pour diviser la surface extérieure au contour proche de D en deux parties. S'assurer que le centre de l'outil est actuellement dans la partie du même côté de la ligne prolongée que le matériau. Sinon, déplacer l'outil dans cette partie. Par exemple, le point E représente la position courante du centre de l'outil. Comme il est du même côté de la ligne FD prolongée que le triangle gris du matériau, aucun mouvement supplémentaire n'est nécessaire. Maintenant écrire la ligne de code NC qui active la compensation et faire le mouvement vers le point D

```
N0010 G41 G1 X2 Y2 (active la compensation et fait le mouvement d'éentre)
```

Cette méthode fonctionnera également avec un angle aigu sur un contour sur parcours d'outil, si l'outil est sur-dimensionné, mais elle échouera si il est sous-dimensionné.




#### 13.5.2.4 Autres points où est exécutée la compensation de rayon d'outil

Le jeu complet de fonctions canoniques comprend des fonctions qui activent et désactivent la compensation de rayon d'outil, de sorte qu'elle puisse être activée quand le contrôleur exécute une de ces fonctions. Dans l'interpréteur cependant, ces commandes ne sont pas utilisées. La compensation est assurée par l'interpréteur et reflétée dans les sorties des commandes, c'est l'interpréteur qui continuera à diriger les mouvements du centre de l'outil. Cela simplifie le travail du contrôleur de mouvement tout en rendant le travail de l'interpréteur un peu plus difficile.

#### 13.5.2.5 Algorithmes pour compensation de rayon d'outil

L'interpréteur permet que les mouvements d'entrée et de sortie soient des arcs. Le comportement pour les mouvements intermédiaires est le même, excepté que certaines situations sont traitées comme des erreurs par l'interpréteur alors qu'elles ne le sont pas sur d'autres contrôleurs de machine.

Données relatives à la compensation de rayon d'outil:

L'interpréteur conserve trois données pour la compensation de rayon d'outil: Le réglage lui même (gauche, droite ou arrêt), program_x et program_y. Les deux dernières représentent les positions en X et en Y données dans le code NC quand la compensation est active. Quand elle est désactivée, les deux entrées sont fixées à de très petites valeurs (10 e-20 ) dont la valeur symbolique (dans un #define) est *unknown*. L'interpréteur utilise, les items current_x et current_y qui représentent, le centre de la pointe de l'outil (dans le système de coordonnées courant), à tout moment.

### 13.5.3 Exemples de Jon Elson

Toutes les informations spécifiques au système se réfèrent au programme LinuxCNC du NIST, mais doit aussi s'appliquer aux plus modernes contrôleurs CNC. Ma méthode de vérification de ces programmes est d'abord de sélectionner l'outil zéro, de sorte que les commandes de compensation soient ignorées. Ensuite, je colle une feuille de papier sur une plaque tenue de niveau dans l'étau, une sorte de platine. J'installe une recharge de stylo à ressort dans la broche. C'est une recharge standard de stylo à bille en métal avec un ressort, dans un corps de 12mm de diamètre. Elle à un ressort pour la faire rentrer dans le corps du stylo, et un *collet* à l'arrière qui permet à la pointe de se rétracter malgré le ressort, mais qui la laisse centrée à quelques dixièmes près. Je charge le programme avec l'outil zéro sélectionné, et il trace une ligne à l'extérieur de la pièce. (voir la figure suivante) Alors, je sélectionne un outil avec le diamètre de l'outil que j'envisage d'utiliser et je lance le programme une nouvelle fois. (Noter que la coordonnée Z dans le programme ne doit pas être changée pour éviter de plonger le stylo au travers du plateau ;-) Maintenant, je dois voir si la compensation G41 ou G42 que je spécifie passe sur le côté voulu de la pièce. Sinon, je modifie avec la compensation du côté opposé, et j'édite la compensation opposée dans le programme, puis j'essaye à nouveau. Maintenant, avec l'outil sur le côté correct de la pièce, je peut vérifier si quelque part sur le parcours l'outil est *trop gros* pour usiner les surfaces concaves. Ma vieille Allen-Bradley 7320 était très indulgente sur ce point, mais LinuxCNC ne tolère rien. Si vous avez la moindre concavité où deux lignes se rencontrent à moins de 180 degrés avec un outil de taille définies, LinuxCNC va s'arrêter là, avec un message

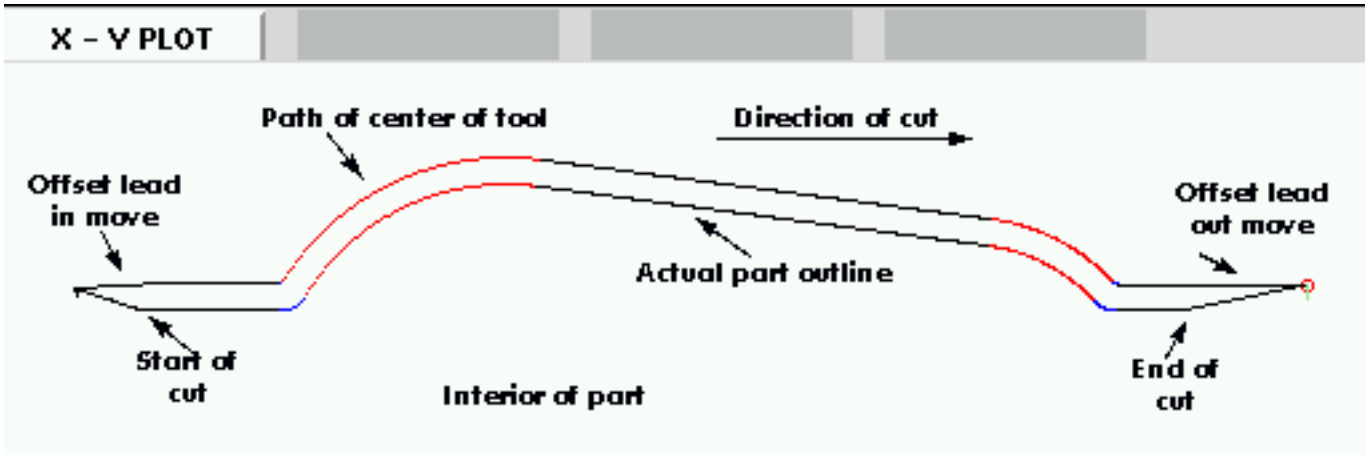
d'erreur. Même si le gougeage est de .001mm de profondeur. Alors, je fais toujours l'approche sur le mouvement d'entrée et le mouvement de sortie juste sur un coin de la pièce, en fournissant un angle de plus de 180 degrés, afin que LinuxCNC ne râle pas. Cela exige une grande attention lors de l'ajustement des points de départ et de sortie, qui ne sont pas compensés par le rayon d'outil, mais ils doivent être choisis avec un rayon approximatif bien réfléchi.

Les commandes sont:
– G40 Annuler la compensation de rayon d'outil
– G41 Activer la compensation, outil à gauche du profil
– G42 Activer la compensation, outil à droite du profil
Voici un petit fichier qui usine le côté d'une pièce avec de multiples arcs convexes et concaves et plusieurs lignes droites. La plupart de ces commandes ont été tracées depuis Bobcad/CAM, mais les lignes N15 et N110 ont été ajoutées par moi et certaines coordonnées dans ce contour ont été bricolées un peu par moi.

```
N10 G01 G40 X-1.3531 Y3.4
N15 F10 G17 G41 D4 X-0.7 Y3.1875 (ligne d'éentre)
N20 X0. Y3.1875
N40 X0.5667 F10
N50 G03 X0.8225 Y3.3307 R0.3
N60 G02 X2.9728 Y4.3563 R2.1875
N70 G01 X7.212 Y3.7986
N80 G02 X8.1985 Y3.2849 R1.625
N90 G03 X8.4197 Y3.1875 R0.3
N100 G01 X9.
N110 G40 X10.1972 Y3.432 (ligne de sortie)
N220 M02
```

La ligne 15 contient G41 D4, qui signifie que le diamètre de l'outil est celui de l'outil #4 dans la table d'outils, il sera utilisé pour décaler la broche de 1/2 diamètre, qui est, bien sûr, le rayon d'outil. Noter que la ligne avec la commande G41 contient le point final du mouvement dans lequel la compensation de rayon est interpolée. Cela signifie qu'au début de ce mouvement, il n'y a aucun effet de compensation et à la fin, l'outil est décalé de 100% du rayon de l'outil sélectionné. Immédiatement après le G41 il y a D4, signifiant que le décalage sera le rayon de l'outil N°4 dans la table d'outils. Noter que les DIAMÈTRES d'outil sont entrés dans la table d'outils. (le diamètre de l'outil de Jon est de 0.4890)

Mais, noter qu'à la ligne 110, où il y a la commande G40, l'interpolation de la compensation d'outil est en dehors de ce mouvement. La manière d'obtenir ce réglage, les mouvements des lignes 15 et 110 sont presque exactement parallèles à l'axe X et la différence dans les coordonnées Y est à la ligne où l'outil est appelé, en dehors de la compensation d'outil.




Certaines autres choses sont à noter, le programme commence avec G40, pour désactiver les compensations éventuellement actives. Cela évite un tas d'ennuis quand le programme s'arrête à cause d'une erreur de concavité, mais laisse la compensation désactivée. Noter aussi, en ligne 15, G17 est utilisé pour spécifier le plan de travail XY pour les interpolations circulaires. J'ai utilisé le format rayon pour les spécifications des arcs plutôt que la forme I, J. LinuxCNC est très pointilleux au sujet des rayons qu'il calcule à partir du format des coordonnées I, J et il doit trouver le début et la fin du mouvement avec 10^-11 unités internes, de sorte qu'il y a beaucoup de problèmes avec des arcs arbitraires. Normalement, si vous avez un arc de 90 degrés, centré sur (1.0,1.0) avec un rayon de 1", tout ira bien, mais si le rayon ne peut pas être exprimé exactement et avec juste le nombre de chiffres significatifs, ou si l'arc à un nombre étrange de degrés, alors les problèmes commencent avec LinuxCNC. Le mot R

supprime tous ce désordre et il est beaucoup plus facile de travailler avec lui, de toute façon. Si l'arc est de plus de 180 degrés, R doit être négatif.

# Chapitre 14

# Vue générale du langage G-codes de LinuxCNC

## 14.1 Brève description du G-code de LinuxCNC

Le G-code est le langage de programmation des machines numériques. Le G-code utilisé par LinuxCNC est basé sur le langage RS274/NGC. Cette documentation le décrit de manière exhaustive, c'est donc un gros morceau mais il contient beaucoup de concepts qui seront assimilés par le lecteur dès la première lecture. C'est notamment le cas de ce chapitre. Par la suite, l'utilisateur reviendra ici, d'abord pour chaque détail de création de son G-code, puis plus tard, seulement pour vérifier la syntaxe des codes les moins courants. Il aura alors perçu la puissance de ce langage et de LinuxCNC qui le met à profit.

## 14.2 Format des paramètres du G-code

Le langage G-code est basé sur des lignes de code. Chaque ligne (également appelée un *bloc*) peut inclure des commandes pour faire produire diverses actions à la machine. Plusieurs lignes de code peuvent être regroupées dans un fichier pour créer un programme G-code.

Une ligne de code typique commence par un numéro de ligne optionnel suivi par un ou plusieurs *mots*. Un mot commence par une lettre suivie d'un nombre (ou quelque chose qui permet d'évaluer un nombre). Un mot peut, soit donner une commande, soit fournir un argument à une commande. Par exemple, *G1 X3* est une ligne de code valide avec deux mots. *G1* est une commande qui signifie *déplaces toi en ligne droite à la vitesse programmée* et *X3* fournit la valeur d'argument (la valeur de X doit être 3 à la fin du mouvement). La plupart des commandes G-code commencent avec une lettre G ou M (G pour Général et M pour Miscellaneous (auxiliaire)). Les termes pour ces commandes sont *G-codes* et *M-codes.*

Le langage G-code n'a pas d'indicateur de début et de fin de programme. L'interpréteur cependant traite les fichiers. Un programme simple peut être en un seul fichier, mais il peut aussi être partagé sur plusieurs fichiers. Un fichier peut être délimité par le signe pour-cent de la manière suivante. La première ligne non vide d'un fichier peut contenir un signe % seul, éventuellement encadré d'espaces blancs, ensuite, à la fin du fichier on doit trouver une ligne similaire. Délimiter un fichier avec des % est facultatif si le fichier comporte un *M2* ou un *M30* , mais est requis sinon. Une erreur sera signalée si un fichier a une ligne pour-cent au début, mais pas à la fin. Le contenu utile d'un fichier délimité par pour-cent s'arrête après la seconde ligne pour-cent. Tout le reste est ignoré.

Le langage G-code prévoit les deux commandes (*M2* ou *M30* ) pour finir un programme. Le programme peut se terminer avant la fin du fichier. Les lignes placées après la fin d'un programme ne seront pas exécutées. L'interpréteur ne les lit pas.

## 14.3 Format d'une ligne

Une ligne de G-code typique est construite de la façon suivante, dans l'ordre avec la restriction à un maximum de 256 caractères sur la même ligne.

1. Un caractère optionnel d'effacement de bloc, qui est la barre oblique */*.

2. Un numéro de ligne optionnel.
3. Un nombre quelconque de mots, valeurs de paramètres et commentaires.
4. Un caractère de fin de ligne (retour chariot ou saut de ligne ou les deux).

Toute entrée non explicitement permise est illégale, elle provoquera un message d'erreur de l'interpréteur.

Les espaces sont permis ainsi que les tabulations dans une ligne de code dont ils ne changent pas la signification, excepté dans les commentaires. Ceci peut donner d'étranges lignes, mais elles sont autorisées. La ligne *g0x +0. 12 34y 7* est équivalente à*g0 x+0.1234 y7*, par exemple.

Les lignes vides sont permises, elles seront ignorées.

La casse des caractères est ignorée, excepté dans les commentaires. Toutes les lettres en dehors des commentaires peuvent être, indifféremment des majuscules ou des minuscules sans changer la signification de la ligne.

## 14.4 Caractère d'effacement de bloc

Le caractère optionnel d'effacement de bloc qui est la barre oblique */*, quand il est placé en premier sur une ligne, peut être utilisé par certaines interfaces utilisateur pour sauter, si besoin, des lignes de code. Dans Axis, la combinaison de touches *Alt-m-/* est une bascule qui active ou désactive l'effacement de bloc. Quand l'effacement de bloc est actif, toutes les lignes commençant par */* sont sautées.

Dans Axis il est également possible de basculer l'activation d'effacement de bloc avec l'icône:

## 14.5 Numéro de ligne

Un numéro de ligne commence par la lettre N suivie d'un nombre entier non signé. Les numéros de ligne peuvent se suivre, être répétés ou être dans le désordre, bien qu'une pratique normale évite ce genre d'usage. Les numéros de ligne peuvent être sautés, c'est une pratique normale. L'utilisation d'un numéro de ligne n'est pas obligatoire, ni même recommandée, mais si ils sont utilisés, il doivent être placés en début de ligne.

## 14.6 Les mots

Un mot est une lettre, autre que N, suivie d'un nombre réel.

Les mots peuvent commencer avec l'une ou l'autre des lettres indiquées dans le tableau ci-dessous. Ce tableau inclus N pour être complet, même si, comme défini précédemment, les numéros de lignes ne sont pas des mots. Plusieurs lettres (I, J, K, L, P, R) peuvent avoir différentes significations dans des contextes différents. Les lettres qui se réfèrent aux noms d'axes ne sont pas valides sur une machine n'ayant pas les axes correspondants.

TABLE 14.1: Les mots et leur signification

| **Lettre** | **Signification** |
|---|---|
| A | Axe A de la machine |
| B | Axe B de la machine |
| C | Axe C de la machine |
| D | Valeur de la compensation de rayon d'outil |
| F | Vitesse d'avance travail |
| G | Fonction Générale (voir la table des codes modaux) |
| H | Index d'offset de longueur d'outil |
| I | Décalage en X pour les arcs et dans les cycles préprogrammés G87 |
| J | Décalage en Y pour les arcs et dans les cycles préprogrammés G87 |

TABLE 14.1: (continued)

| Lettre | Signification |
|---|---|
| K | Décalage en Z pour les arcs et dans les cycles préprogrammés G87 |
| | Distance de déplacement par tour de broche avec G33 |
| M | Fonction auxiliaire (voir la table des codes modaux) |
| N | Numéro de ligne |
| P | Temporisation utilisée dans les cycles de perçage et avec G4. |
| | Mot clé utilisé avec G10. |
| Q | Incrément Delta en Z dans un cycle G73, G83 |
| R | Rayon d'arc ou plan de retrait dans un cycle préprogrammé |
| S | Vitesse de rotation de la broche |
| T | Numéro d'outil |
| U | Axe U de la machine |
| V | Axe V de la machine |
| W | Axe W de la machine |
| X | Axe X de la machine |
| Y | Axe Y de la machine |
| Z | Axe Z de la machine |

## 14.7 Les nombres

Les règles suivantes sont employées pour des nombres (explicites). Dans ces règles un chiffre est un caractère simple entre 0 et 9.

- Un nombre commence par:
  - un signe plus ou un signe moins optionnel, suivi de
  - zéro à plusieurs chiffres, peut être suivis par,
  - un point décimal, suivi de
  - zéro à plusieurs chiffres, il doit au moins y avoir un chiffre.
- Il existe deux types de nombres:
  - Les entiers, qui n'ont pas de point décimal.
  - Les décimaux, qui ont un point décimal.
- Les nombres peuvent avoir n'importe quel nombre de chiffres, sous réserve de la limitation de longueur d'une ligne. Seulement environ dix-sept chiffres significatifs seront retenus, c'est toutefois suffisant pour toutes les applications connues.
- Un nombre non nul sans autre signe que le premier caractère est considéré positif.

Les zéros non significatifs, ne sont pas nécessaires.

Si un nombre utilisé dans le langage G-code est proche d'une valeur entière à moins de quatre décimales, il est considéré comme entier, par exemple 0.9999.

## 14.8 Paramètres (Variables)

Le langage RS274/NGC supporte les *paramètres*, qui sont appelés *variables* dans d'autres langages de programmation. Il existe plusieurs types de paramètres ayant différents usages et différentes formes. Le seul type de nombre supporté par les paramètres est le flottant, il n'y a pas de string, pas de boolean ni d'entier dans le G-code comme dans d'autres langages de programmation. Toutefois, les expressions logiques peuvent être formulées avec les opérateurs booléens (*AND*, *OR*, *XOR* et les opérateurs de comparaison *EQ*, *NE*, *GT*, *GE*, *LT*, *LE*) ainsi que *MOD*, *ROUND*, *FUP* et *FIX* les fonctions qui supportent l'arithmétique entière.

Les paramètres différent par leur syntaxe, leur portée, leur comportement quand ils ne sont pas encore initialisés, leur mode, leur persistance et l'usage pour lequel ils sont prévus.

**Syntaxes**

Il y a trois sortes d'apparences syntaxiques:

- *numéroté* - #4711
- *nommé local* - #<valeurlocale>
- *nommé global* - #<_valeurglobale>

**La portée**

La portée d'un paramètre est soit globale, ou locale à l'intérieur d'un sous-programme. Les paramètres de sous-programme et les paramètres nommés ont une portée locale. Les paramètres nommés globaux et les paramètres numérotés commencent par un nombre, exemple: 31 a une portée globale. RS274/NGC utilise une *portée lexicale*, dans un sous-programme, seules sont locales les variables qui y sont définies et toutes les variables globales y sont visibles. Les variables locales à un appel de procédure, ne sont pas visibles dans la procédure appelée.

**Le comportement des paramètres non encore initialisés**

1. Les paramètres globaux non initialisés et les paramètres de sous-programmes inutilisés, retournent la valeur zéro quand ils sont utilisés dans une expression.
2. Les paramètres nommés signalent une erreur quand ils sont utilisés dans une expression.

**Le mode**

La plupart des paramètres sont en lecture/écriture et peuvent être assignés dans une instruction d'affectation. Cependant, pour beaucoup de paramètres prédéfinis, cela n'a pas de sens, ils sont alors en lecture seule. Ils peuvent apparaître dans les expressions, mais pas sur le côté gauche d'une instruction d'affectation.

**La persistance**

Quand LinuxCNC s'arrête, les paramètres volatiles perdent leurs valeurs. Tous les paramètres sont volatiles, excepté les paramètres numérotés dans l'étendue courante de persistance [1]. Les paramètres persistants sont enregistrés dans un fichier *.var* et restaurés à leurs valeurs précédentes quand LinuxCNC est relancé. Les paramètres numérotés volatiles sont remis à zéro.

**Utilisation prévue**

1. Paramètres utilisateur:: paramètres numérotés dans l'étendue 31 à 5000, paramètres nommés globaux et locaux excepté les paramètres prédéfinis. Sont disponibles pour une utilisation générale de stockage de valeurs flottantes, comme des résultats intermédiaires, des drapeaux, etc. durant l'exécution d'un programme. Ils sont en lecture/écriture (une valeur peut leur être attribuée).
2. Paramètres de sous-programme - Ils sont utilisés pour conserver les paramètres actuels passés à un sous-programme.
3. paramètres numérotés - la plupart de ces paramètres sont utilisés pour accéder aux offsets des systèmes de coordonnées.
4. paramètres nommés prédéfinis - utilisés pour déterminer l'état de l'interpréteur et de la machine, par exemple *#<_relative>* retourne 1 si G91 est actif et 0 si G90 est activé. Ils sont en lecture seule.

## 14.9 Paramètres numérotés

Un paramètre numéroté commence par le caractère # suivi par un entier compris entre 1 et (actuellement) 5602. Le paramètre est référencé par cet entier, sa valeur est la valeur stockée dans le paramètre.

Une valeur est stockée dans un paramètre avec l'opérateur = par exemple:

```
#3 = 15 (la valeur 15 est éstocke dans le èparamtre énumro 3)
```

Le caractère # a une précédence supérieure à celle des autres opérations, ainsi par exemple, *#1+2* signifie la valeur trouvée en ajoutant 2 à la valeur contenue dans le paramètre 1 et non la valeur trouvée dans le paramètre 3. Bien sûr, *#[1+2]* signifie la valeur trouvée dans le paramètre 3. Le caractère # peut être répété, par exemple *##2* signifie le paramètre dont le numéro est égal à la valeur entière trouvée dans le paramètre 2.

**31 à 5000**

Paramètres des G-Code utilisateur. Ces paramètres sont globaux dans le fichier G-code.

1. L'étendue de persistance courante des paramètres évolue en même temps qu'évolue le développement. Cette étendue est actuellement de 5161 à 5390. Elle est définie par *_required_parameters array* dans le fichier src/linuxcnc/rs274ngc/interp_array.cc .

**5061 à 5069**
Résultat du palpage *G38.2* pour X Y Z A B C U V W. Volatile.

**5161 à 5169**
Coordonnées d'un *G28* pour X Y Z A B C U V W. Persistant.

**5181 à 5189**
Origine *G30* pour X Y Z A B C U V W. Persistant.

**5211 à 5219**
Offset *G92* pour X Y Z A B C U V W. Persistant.

**5220**
Système de coordonnées 1 à 9 pour G54 à G59.3. Persistant.

**5221 à 5229**
Système de coordonnées 1, G54 pour X Y Z A B C U V W R. Persistant.

**5241 à 5249**
Système de coordonnées 2, G55 pour X Y Z A B C U V W R. Persistant.

**5261 à 5269**
Système de coordonnées 3, G56 pour X Y Z A B C U V W R. Persistant.

**5281 à 5289**
Système de coordonnées 4, G57 pour X Y Z A B C U V W R. Persistant.

**5301 à 5309**
Système de coordonnées 5, G58 pour X Y Z A B C U V W R. Persistant.

**5321 à 5329**
Système de coordonnées 6, G59 pour X Y Z A B C U V W R. Persistant.

**5341 à 5349**
Système de coordonnées 7, G59.1 pour X Y Z A B C U V W R. Persistant.

**5361 à 5369**
Système de coordonnées 8, G59.2 pour X Y Z A B C U V W R. Persistant.

**5381 à 5389**
Système de coordonnées 9, G59.3 pour X Y Z A B C U V W R. Persistant.

**5399**
Résultat de M66 - Surveille ou attends une entrée. Volatile.

**5400**
Numéro de l'outil courant. Volatile.

**5401 à 5409**
Offset d'outil pour X Y Z A B C U V W. Volatile.

**5410**
Diamètre de l'outil courant. Volatile.

**5411**
Angle frontal de l'outil courant. Volatile.

**5412**
Angle arrière de l'outil courant. Volatile.

**5413**
Orientation de l'outil. Volatile.

**5420 à 5428**
Positions courantes incluant les offsets, dans l'unité courante du programme pour X Y Z A B C U V W.

## 14.10 Paramètres de sous-programme

– *1-30* - Paramètres d'appel d'arguments, locaux au sous-programme. Voir la section des O-codes.

## 14.11 Paramètres nommés

Les paramètres nommés fonctionnent comme les paramètres numérotés mais sont plus faciles à lire. Les paramètres nommés sont convertis en minuscules, les espaces et tabulations sont supprimés. Les paramètres nommés doivent être encadrés des signes < et >.

*#<Un paramètre nommé>* est un paramètre nommé local. Par défaut, un paramètre nommé est local à l'étendue dans laquelle il est assigné. L'accès à un paramètre local, en dehors de son sous-programme est impossible, de sorte que deux sous-programmes puissent utiliser le même nom de paramètre sans craindre qu'un des deux n'écrase la valeur de l'autre.

*#<_un paramètre global>* est un paramètre nommé global. Ils sont accessibles depuis des sous-programmes appelés et peuvent placer des valeurs dans tous les sous-programmes accessibles à l'appelant. En ce qui concerne la portée, ils agissent comme des paramètres numérotés. Ils ne sont pas enregistrés dans des fichiers.

Exemples:
– Déclaration d'une variable nommée globale

```
#<_troisdents_dia> = 10.00
```

– Référence à la variable globale précédemment déclarée

```
#<_troisdents_rayon> = [#<_troisdents_dia>/2.0]
```

– Mélange de paramètres nommés et de valeurs littérales

```
o100 call [0.0] [0.0] [#<_interieur_decoupe>-#<_troisdents_dia>][#<_Zprofondeur>] [#< ↩
    _vitesse>]
```

## 14.12 Paramètres nommés prédéfinis

Les paramètres globaux suivants sont disponibles en lecture seule, pour accéder aux états internes de l'interpréteur et de la machine. Ils peuvent être utilisés dans les expressions quelconques, par exemple pour contrôler le flux d'un programme avec les instructions *if-then-else*.
– *#<_vmajor>* - Version majeure de LinuxCNC. Si la version courante est 2.5.2, 2.5 est retourné.
– *#<_vminor>* - Version mineure du LinuxCNC. Si la version courante est 2.6.2, 0.2 est retourné.
– *#<_line>* - Numéro de séquence. Si un fichier G-code est en cours, le numéro de la ligne courante est retourné.
– *#<_motion_mode>* - Retourne le mode mouvement courant de l'interpréteur:

| **Mode mouvement** | **Valeur retournée** |
|---|---|
| G1 | 10 |
| G2 | 20 |
| G3 | 30 |
| G33 | 330 |
| G38.2 | 382 |
| G38.3 | 383 |
| G38.4 | 384 |
| G38.5 | 385 |
| G5.2 | 52 |
| G73 | 730 |
| G76 | 760 |
| G80 | 800 |
| G81 | 810 |
| G82 | 820 |
| G83 | 830 |
| G84 | 840 |
| G85 | 850 |

| Mode mouvement | Valeur retournée |
|---|---|
| G86 | 860 |
| G87 | 870 |
| G88 | 880 |
| G89 | 890 |

– *#<_plane>* - Retourne une valeur désignant le plan courant:

| Plan | Valeur retournée |
|---|---|
| G17 | 170 |
| G18 | 180 |
| G19 | 190 |
| G17.1 | 171 |
| G18.1 | 181 |
| G19.1 | 191 |

– *#<_ccomp>* - Statut de la compensation d'outil. Retourne une valeur:

| Mode | Valeur retournée |
|---|---|
| G40 | 400 |
| G41 | 410 |
| G41.1 | 411 |
| G41 | 410 |
| G42 | 420 |
| G42.1 | 421 |

– *#<_metric>* - Retourne 1 si G21 est *on*, sinon 0.
– *#<_imperial>* - Retourne 1 si G20 est *on*, sinon 0.
– *#<_absolute>* - Retourne 1 si G90 est *on*, sinon 0.
– *#<_incremental>* - Retourne 1 si G91 est *on*, sinon 0.
– *#<_inverse_time>* - Retourne 1 si le mode inverse du temps (G93) est *on*, sinon 0.
– *#<_units_per_minute>* - Retourne 1 si le mode unités par minute (G94) est *on*, sinon 0.
– *#<_units_per_rev>* - Retourne 1 si le mode Unités par tour (G95) est *on*, sinon 0.
– *#<_coord_system>* - Retourne l'index du système de coordonnées courant (G54 à G59.3).

| Mode | Valeur retournée |
|---|---|
| G54 | 0 |
| G55 | 1 |
| G56 | 2 |
| G57 | 3 |
| G58 | 4 |
| G59 | 5 |
| G59.1 | 6 |
| G59.2 | 7 |
| G59.3 | 8 |

– *#<_tool_offset>* - Retourne 1 si l'offset d'outil (G43) est *on*, sinon 0.
– *#<_retract_r_plane>* - Retourne 1 si G98 est actif, sinon 0.

– *#<_retract_old_z>* - Retourne 1 si G99 est *on*, sinon 0.

## 14.13 Paramètres système

– `#<_spindle_rpm_mode>` - Retourne 1 si la broche est en mode tr/mn (G97), sinon 0.
– `#<_spindle_css_mode>` - Retourne 1 si la broche est en mode vitesse de coupe constante (G96), sinon 0.
– `#<_ijk_absolute_mode>` - Retourne 1 si le mode de déplacement en arc est absolu (G90.1), sinon 0.
– `#<_lathe_diameter_mode>` - Retourne 1 pour un tour configuré en mode diamètre (G7), sinon 0.
– `#<_lathe_radius_mode>` - Retourne 1 pour un tour configuré en mode rayon (G8) , sinon 0.
– `#<_spindle_on>` - Retourne 1 si la broche tourne (M3 ou M4 en cours), sinon 0.
– `#<_spindle_cw>` - Retourne 1 si la broche est dans le sens horaire (M3) sinon 0.
– `#<_mist>` - Retourne 1 si l'arrosage par gouttelettes est activé (M7).
– `#<_flood>` - Retourne 1 si l'arrosage fluide est activé (M8).
– `#<_speed_override>` - Retourne 1 si un correcteur de vitesse d'avance travail est activé (M48 ou M50 P1), sinon 0.
– `#<_feed_override>` - Retourne 1 si un correcteur de vitesse broche est activé (M48 ou M51 P1), sinon 0.
– `#<_adaptive_feed>` - Retourne 1 si un correcteur de vitesse adaptative est activé (M52 ou M52 P1), sinon 0.
– `#<_feed_hold>` - Retourne 1 si le contrôle de coupure vitesse est activé (M53 P1), sinon 0.
– `#<_feed>` - Retourne la valeur courante d'avance travail (F).
– `#<_rpm>` - Retourne la valeur courante de vitesse broche (S).
– `#<_x>` - Retourne la coordonnée machine courante en X. Identique à #5420.
– `#<_y>` - Retourne la coordonnée machine courante en Y. Identique à #5421.
– `#<_z>` - Retourne la coordonnée machine courante en Z. Identique à #5422.
– `#<_a>` - Retourne la coordonnée machine courante en A. Identique à #5423.
– `#<_b>` - Retourne la coordonnée machine courante en B. Identique à #5424.
– `#<_c>` - Retourne la coordonnée machine courante en C. Identique à #5425.
– `#<_u>` - Retourne la coordonnée machine courante en U. Identique à #5426.
– `#<_v>` - Retourne la coordonnée machine courante en V. Identique à #5427.
– `#<_w>` -Retourne la coordonnée machine courante en W. Identique à #5428.
– `#<_current_tool>` - Retourne le N° de l'outil courant monté dans la broche. Identique à #5400.
– `#<_current_pocket>` - Retourne le N° de poche de l'outil courant.
– `#<_selected_tool>` - Retourne le N° de l'outil sélectionné par le mot T. Par défaut -1.
– `#<_selected_pocket>` - Retourne le N° de poche sélectionné par le mot T. Par défaut -1 (pas de poche sélectionnée).
– `#<_value>` - Retourne la valeur du dernier O-code `return` ou `endsub`. Valeur 0 par défaut si pas d'expression après `return` ou `endsub`. Initialisé à 0 au démarrage du programme.
– `#<_value_returned>` - 1.0 si le dernier O-code `return` ou `endsub` a retourné une valeur, 0 autrement. Effacé par le prochain appel à un O-code.
– `#<_task>` - 1.0 si l'instance en cours d'exécution par l'interpréteur fait partie d'une tâche de fraisage, 0.0 autrement. Il est parfois nécessaire de traiter ce cas particulier pour conserver un chemin d'outil propre, par exemple quand on teste le succès d'une mesure au palpeur (G38.x), en examinant #5070, ce qui ratait toujours dans le chemin d'outil de l'interpréteur (ex: Axis).
– `#<_call_level>` - current nesting level of O-word procedures. Pour débogage.
– `#<_remap_level>` - current level of the remap stack. Each remap in a block adds one to the remap level. Pour débogage.

## 14.14 Expressions

Une expression est un groupe de caractères commençant avec le crochet gauche *[* et se terminant avec le crocher droit *]* . Entre les crochets, on trouve des nombres, des valeurs de paramètre, des opérations mathématiques et d'autres expressions. Une expression est évaluée pour produire un nombre. Les expressions sur une ligne sont évaluées quand la ligne est lue et avant que quoi que ce soit ne soit éxécuté sur cette ligne. Un exemple d'expression: *[1 + acos[0] - [#3 ** [4.0/2]]]*.

## 14.15 Opérateurs binaires

Les opérateurs binaires ne se rencontrent que dans les expressions. Il y a quatre opérateurs mathématiques de base: addition +, soustraction -, multiplication * et division /. Il y a trois opérateurs logiques: le *ou (OR)*, le *ou exclusif (XOR)* et le *et logique*

*(AND)*. Le huitième opérateur est le *modulo (MOD)*. Le neuvième opérateur est l'élévation à la puissance *(**)* qui élève le nombre situé à sa gauche à la puissance du nombre situé à sa droite. Les opérateurs de relation sont: égalité *(EQ)*, non égalité *(NE)*, strictement supérieur *(GT)*, supérieur ou égal *(GE)*, strictement inférieur *(LT)* et inférieur ou égal *(LE)*.

Les opérations binaires sont divisées en plusieurs groupes selon leur précédence. Si dans une opération se trouvent différents groupes de précédence, par exemple dans l'expression *[2.0 / 3 * 1.5 - 5.5 / 11.0]*, les opérations du groupe supérieur seront effectuées avant celles des groupes inférieurs. Si une expression contient plusieurs opérations du même groupe (comme les premiers / et * dans l'exemple), l'opération de gauche est effectuée en premier. Notre exemple est équivalent à: *[[[2.0/3]*1.5]-[5.5/11.0]]*, qui est équivalent à *[1.0-0.5]* , le résultat est: *0.5* .

Les opérations logiques et le modulo sont exécutés sur des nombres réels et non pas seulement sur des entiers. Le zéro est équivalent à un état logique faux (FALSE), tout nombre différent de zéro est équivalent à un état logique vrai (TRUE).

**Précédence des opérateurs**

| **Opérateurs** | **Précédence** |
|---|---|
| ** | *haute* |
| * / MOD | |
| + - | |
| EQ NE GT GE LT LE | |
| AND OR XOR | *basse* |

## 14.16 Fonctions

Une fonction commence par son nom, ex: *ATAN* suivi par une expression divisée par une autre expression (par exemple *ATAN[2]/[1+3]*) ou tout autre nom de fonction suivi par une expression (par exemple *SIN[90]*). Les fonctions disponibles sont visibles le tableau ci-dessous. Les arguments pour les opérations unaires sur des angles ( *COS*, *SIN* et *TAN* ) sont en degrés. Les valeurs retournées par les opérations sur les angles ( *ACOS*, *ASIN* et *ATAN* ) sont également en degrés.

La fonction *FIX* arrondi un nombre vers la gauche, (moins positif ou plus négatif) par exemple, *FIX[2.8]=2* et *FIX[-2.8]=-3*. La fonction *FUP* à l'inverse, arrondi un nombre vers la droite (plus positif ou moins négatif) par exemple, *FUP[2.8]=3* et *FUP[-2.8]=-2*.

La fonction *EXISTS* vérifie l'existence d'un simple paramètre nommé. Il reçoit le paramètre à vérifier en argument, il retourne 1 si celui-ci existe et 0 sinon. C'est une erreur si un paramètre numéroté ou une expression est utilisé.

TABLE 14.2: Fonctions

| **Nom de fonction** | **Fonction** |
|---|---|
| ATAN[Y]/[X] | Tangente quatre quadrants |
| ABS[arg] | Valeur absolue |
| ACOS[arg] | Arc cosinus |
| ASIN[arg] | Arc sinus |
| COS[arg] | Cosinus |
| EXP[arg] | Exposant |
| FIX[arg] | Arrondi à l'entier immédiatement inférieur |
| FUP[arg] | Arrondi à l'entier immédiatement supérieur |
| ROUND[arg] | Arrondi à l'entier le plus proche |
| LN[arg] | Logarithme Néperien |
| SIN[arg] | Sinus |
| SQRT[arg] | Racine carrée |
| TAN[arg] | Tangente |
| EXISTS[arg] | Vérifie l'existence d'un paramètre nommé |

## 14.17 Répétitions d'items

Une ligne peut contenir autant de mots G que voulu, mais un seul du même groupe modal.

Une ligne peut avoir de zéro à quatre mots M. Mais pas deux mots M du même groupe modal.

Pour toutes les autres lettres légales, un seul mot commençant par cette lettre peut se trouver sur la même ligne.

Si plusieurs valeurs de paramètre se répètent sur la même ligne, par exemple: *#3=15 #3=6*, seule la dernière valeur prendra effet. Il est absurde, mais pas illégal, de fixer le même paramètre deux fois sur la même ligne.

Si plus d'un commentaire apparaît sur la même ligne, seul le dernier sera utilisé, chacun des autres sera lu et son format vérifié, mais il sera ignoré. Placer plusieurs commentaires sur la même ligne est très rare.

## 14.18 Ordre des items

Les trois types d'item dont la commande peut varier sur une ligne (comme indiqué au début de cette section) sont les mots, les paramètres et les commentaires. Imaginez que ces trois types d'éléments sont divisés en trois groupes selon leur type.

Dans le premier groupe les mots, peuvent être arrangés dans n'importe quel ordre sans changer la signification de la ligne.

Dans le second groupe les valeurs de paramètre, quelque soit leur arrangement, il n'y aura pas de changement dans la signification de la ligne sauf si le même paramètre est présent plusieurs fois. Dans ce cas, seule la valeur du dernier paramètre prendra effet. Par exemple, quand la ligne *#3=15 #3=6* aura été interprétée, la valeur du paramètre 3 vaudra 6. Si l'ordre est inversé, *#3=6 #3=15* après interprétation, la valeur du paramètre 3 vaudra 15.

Enfin dans le troisième groupe les commentaires, si plusieurs commentaires sont présents sur une ligne, seul le dernier commentaire sera utilisé.

Si chaque groupe est laissé, ou réordonné, dans l'ordre recommandé, la signification de la ligne ne changera pas, alors les trois groupes peuvent être entrecroisés n'importe comment sans changer la signification de la ligne. Par exemple, la ligne *g40 g1 #3=15 (foo) #4=-7.0* à cinq items est signifiera exactement la même chose dans les 120 ordres d'arrangement possibles des cinq items comme *#4=-7.0 g1 #3=15 g40 (foo)*.

## 14.19 Commandes et modes machine

En G-code, de nombreuses commandes produisent, d'un mode à un autre, quelque chose de différent au niveau de la machine, le mode reste actif jusqu'à ce qu'une autre commande ne le révoque, implicitement ou explicitement. Ces commandes sont appelées *modales*. Par exemple, si l'arrosage est mis en marche, il y reste jusqu'à ce qu'il soit explicitement arrêté. Les G-codes pour les mouvements sont également modaux. Si, par exemple, une commande G1 (déplacement linéaire) se trouve sur une ligne, elle peut être utilisée sur la ligne suivante avec seulement un mot d'axe, tant qu'une commande explicite est donnée sur la ligne suivante en utilisant des axes ou un arrêt de mouvement.

Les codes *non modaux* n'ont d'effet que sur la ligne ou ils se présentent. Par exemple, G4 (tempo) est non modale.

## 14.20 Coordonnées polaires

Des coordonnées polaires peuvent être utilisées pour spécifier les coordonnées *XY* d'un mouvement. Le *@n* est la distance et le *^n* est l'angle. L'avantage est important, par exemple: Pour faire très simplement un cercle de trous tangents:
- Passer un point situé au centre du cercle
- Régler la compensation de longueur d'outil
- Déplacer l'outil vers le premier trou
- Enfin, lancer le cycle de perçage.

Les coordonnées polaires sont toujours données à partir de la position *X0, Y0*. Pour décaler les coordonnées polaires machine utilisez le décalage pièce ou sélectionnez un système de coordonnées.

En mode absolu, la distance et l'angle sont donnés à partir de la position *X0, Y0* et l'angle commence à *0* sur l'axe X positif et augmente dans la direction trigonométrique (anti-horaire) autour de l'axe Z. Le code `G1 @1 ^90` est la même que `G1 Y1`.

En mode relatif, la distance et l'angle sont également donnés à partir de la position *XY zéro*, mais ils sont cumulatifs. Ce fonctionnement en mode incrémental peut être déroutant au début.

Par exemple: si vous avez le programme suivant, vous vous attendez à obtenir une trajectoire carré.

```
F100 G1 @.5 ^90
G91 @.5 ^90
@.5 ^90
@.5 ^90
@.5 ^90
G90 G0 X0 Y0 M2
```

Vous pouvez voir sur la figure suivante que la sortie n'est pas celle à laquelle vous vous attendiez, parce-que avons ajouté 0.5 à la distance de la position XY zéro à chaque début de ligne.

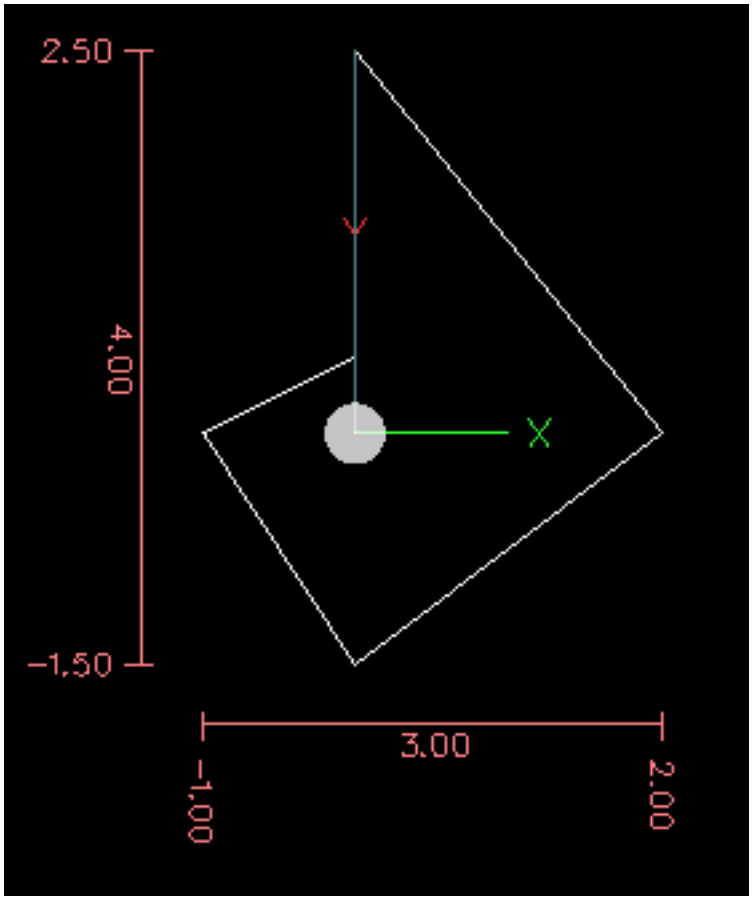



FIGURE 14.1 – Spirale polaire

Le code suivant va produire notre modèle carré.

```
F100 G1 @.5 ^90
G91 ^90
^90
^90
^90
```

```
G90 G0 X0 Y0 M2
```

Comme vous pouvez le voir, en ajoutant seulement l'angle de 90 degrés à chaque ligne. La distance du point final est la même pour chaque ligne.

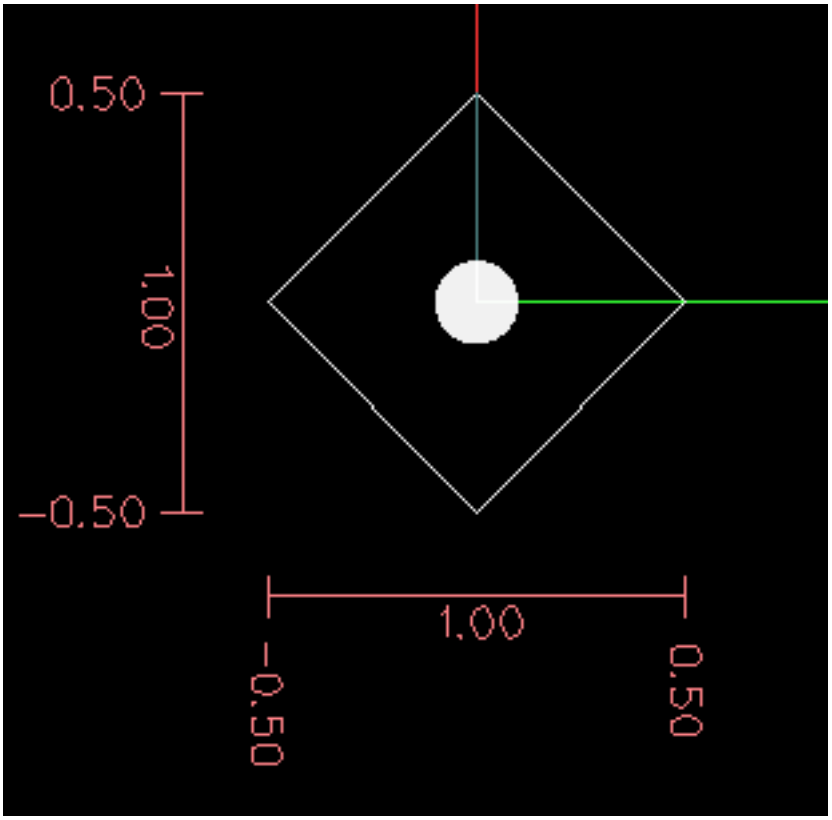



FIGURE 14.2 – Carré polaire

C'est une erreur si:
- Un mouvement incrémental est lancé à l'origine.
- Un mélange de mots polaires et de X ou Y est utilisé.

## 14.21 Groupes modaux

Les commandes modales sont arrangées par lots appelés *groupes modaux*, à tout moment, un seul membre d'un groupe modal peut être actif. En général, un groupe modal contient des commandes pour lesquelles il est logiquement impossible que deux membres soient actifs simultanément, comme les unités en pouces et les unités en millimètres. Un centre d'usinage peut être dans plusieurs modes simultanément, si un seul mode pour chaque groupe est actif. Les groupes modaux sont visibles dans le tableau ci-dessous.

**Groupes modaux des G-codes**

| Signification du groupe modal | Mots G |
|---|---|
| Codes non modaux (*Groupe 0*) | G4, G10, G28, G30, G53, G92, G92.1, G92.2, G92.3 |
| Mouvements (*Groupe 1*) | G0, G1, G2, G3, G33, G38.x, G73, G80, G81, |
| | G82, G83, G84, G85, G86, G87, G88, G89 |
| Choix du plan de travail (*Groupe 2*) | G17, G18, G19, G17.1, G18.1, G19.1 |
| Mode déplacement (*Groupe 3*) | G90, G91 |
| Mode déplacement en arc IJK (*Groupe 4*) | G90.1, G91.1 |

| Signification du groupe modal | Mots G |
|---|---|
| Mode de vitesses (*Groupe 5*) | G93, G94, G95 |
| Unités machine (*Groupe 6*) | G20, G21 |
| Compensation de rayon d'outil (*Groupe 7*) | G40, G41, G42, G41.1, G42.1 |
| Compensation de longueur d'outil (*Groupe 8*) | G43, G43.1, G49 |
| Plan de retrait cycle de perçage (*Groupe 10*) | G98, G99 |
| Systèmes de coordonnées (*Groupe 12*) | G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3 |
| Mode contrôle de trajectoire (*Groupe 13*) | G61, G61.1, G64 |
| Mode contrôle vitesse broche (*Groupe 14*) | G96, G97 |
| Mode diamètre/rayon sur les tours (*Groupe 15*) | G7, G8 |

**Groupes modaux des M-codes**

| Signification du groupe modal | Mots M |
|---|---|
| Types de fin de programme (*Groupe 4*) | M0, M1, M2, M30, M60 |
| On/Off I/O (*Groupe 5*) | M6 Tn |
| Appel d'outil (*Groupe 6*) | M6 Tn |
| Commande de broche (*Groupe 7*) | M3, M4, M5, M19 |
| Arrosages (*Groupe 8*) | (M7, M8, peuvent être actifs simultanément), M9 |
| Boutons de correction de vitesse (*Groupe 9*) | M48, M49, M50, M51 |
| Définis par l'utilisateur (*Groupe 10*) | M100 à M199 |

Pour plusieurs groupes modaux, quand la machine est prête à accepter des commandes, un membre du groupe doit être en vigueur. Il y a des paramètres par défaut pour ces groupes modaux. Lorsque la machine est mise en marche ou ré-initialisées, les valeurs par défaut sont automatiquement actives.

Groupe 1, le premier groupe du tableau, est un groupe de G-codes pour les mouvements. À tout moment, un seul d'entre eux est actif. Il est appelé le mode de mouvement courant.

C'est une erreur que de mettre un G-code du groupe 1 et un G-code du groupe 0 sur la même ligne si les deux utilisent les mêmes axes. Si un mot d'axe utilisant un G-code du groupe 1 est implicitement actif sur la ligne (en ayant été activé sur une ancienne ligne) et qu'un G-code du groupe 0 utilisant des mots d'axes apparaît sur la même ligne, l'activité du G-code du groupe 1 est révoquée pour le reste de la ligne. Les mots d'axes utilisant des G-codes du groupe 0 sont G10, G28, G30 et G92.

C'est une erreur d'inclure des mots sans rapport sur une ligne avec le contrôle de flux *O*.

## 14.22 Commentaires

Des commentaires peuvent être ajoutés aux lignes de G-code pour clarifier l'intention du programmeur. Les commentaires peuvent être placés sur une ligne en les encadrant par des parenthèses. Ils peuvent aussi occuper tout le reste de la ligne à partir d'un point virgule. Le point virgule n'est pas traité comme un début de commentaire si il se trouve entre deux parenthèses.

Voici un exemple de programme commenté:

```
G0 (Rapide àé dmarrer.) X1 Y1
G0 X1 Y1 (Rapide àé dmarrer; mais n'oubliez pas l'arrosage.)
M2 ; Fin du programme.
```

Les commentaires peuvent se trouver entre des mots, mais pas entre des mots et leur paramètre correspondant. Ainsi, cette ligne est correcte:

```
S100(vitesse broche)F200(vitesse d'avance)
```

mais celle-ci est incorrecte:

```
S(speed)100F(feed)200
```

Les commentaires sont seulement informatifs, ils n'ont aucune influence sur la machine.

Il y a plusieurs commentaires *actif* qui ressemblent à un commentaire mais qui produit certaines actions, comme *(debug,...)* ou *(print,...)*, expliqués plus loin. Si plusieurs commentaires se trouvent sur la même ligne, seul le dernier sera interprété selon les règles. Par conséquent, un commentaire normal suivant un commentaire actif aura pour effet de désactiver le commentaire actif. Par exemple, *(foo) (debug,#1)* affichera la valeur du paramètre *#1*, mais *(debug,#1) (foo)* ne l'affichera pas.

Un commentaire commençant par un point virgule est par définition le dernier commentaire sur cette ligne et sera toujours interprété selon la syntaxe des commentaires actifs.

## 14.23 Messages

– *(MSG,)* - Un commentaire contient un message si *MSG* apparaît après la parenthèse ouvrante et avant tout autre caractère. Les variantes de *MSG* qui incluent un espace blanc et des minuscules sont permises. Le reste du texte avant la parenthèse fermante est considéré comme un message. Les messages sont affichés sur la visu de l'interface utilisateur.

**Exemple de message**

```
(MSG, Ceci est un message)
```

## 14.24 Enregistrement des mesures

– *(PROBEOPEN filename.txt)* - ouvrira le fichier *filename.txt* et y enregistrera les 9 coordonnées de XYZABCUVW pour chacune des mesures réussie.
– *(PROBECLOSE)*. - fermera le fichier de log palpeur.

Voir la section sur la mesure au palpeur pour d'autres informations sur le palpage avec G38.

## 14.25 Log général

– *(LOGOPEN,filename.txt)* - Ouvre le fichier de log *filename.txt*. Si le fichier existe déjà, il sera tronqué.
– *(LOGAPPEND,filename.txt)* - Ouvre le fichier de log *filename.txt*. Si le fichier existe déjà, il sera ajoutées.
– *(LOGCLOSE)* - Si le fichier est ouvert, il sera fermé.
– *(LOG,message)* - Le *message* placé derrière la virgule est écrit dans le fichier de log si il est ouvert. Supporte l'extension des paramètres comme décrit plus loin.

## 14.26 Messages de débogage

– *(DEBUG,commentaire)* sont traités de la même façon que ceux avec *(msg,reste du commentaire)* avec l'ajout de possibilités spéciales pour les paramètres, comme décrit plus loin.
– *(PRINT,commentaire)* vont directement sur la sortie *stderr* avec des possibilités spéciales pour les paramètres, comme décrit plus loin.

## 14.27 Paramètres dans les commentaires

Dans les commentaires avec DEBUG, PRINT et LOG, les valeurs des paramètres dans le message sont étendues.

Par exemple: pour afficher le contenu d'une variable nommée globale sur la sortie stderr (la fenêtre de la console par défaut), ajouter une ligne au G-code comme:

**Exemple de paramètres en commentaire**

```
(print,èdiamtre fraise 3 dents = #<_troisdents_dia>)
(print,la valeur de la variable 123 est: #123)
```

À l'intérieur de ces types de commentaires, les séquences comme *123* sont remplacées par la valeur du paramètre 123. Les séquences comme <*paramètre nommé*> sont remplacées par la valeur du paramètre nommé. Rappelez vous que les espaces dans les noms des paramètres nommés sont supprimés, <*parametre nomme*> est équivalent à <*parametrenomme*>.

## 14.28 Exigences des fichiers

Un programme G-code doit contenir une ou plusieurs lignes de G-code puis se terminer par une ligne defin de programme. Tout G-code, placé après cette ligne de fin de programme, sera ignoré.

Si le programme n'utilise pas G-code de fin de programme, une paire de signes pourcent % peut être utilisées. Le premier signe % doit dans ce cas se trouver sur la première ligne du fichier, suivi par une ou plusieurs lignes de G-code, puis du second signe %. Tout G-code placé après le second signe % sera ignoré.

**Note**
Les fichiers de G-code doivent être créés avec un éditeur de texte comme Gedit et non avec un traitement de texte comme Open Office. Les traitements de texte ajoutent de nombreux caractères de contrôle dans les fichiers, ce qui les rends inutilisables comme programmes G-code.

## 14.29 Taille des fichiers

L'interpréteur et le gestionnaire de tâches ont été écrits, de sorte que la taille des fichiers n'est limité que par la capacité du disque dur. Les interfaces graphiques TkLinuxCNC et Axis affichent tous les deux le programme G-code à l'écran pour l'utilisateur, cependant, la RAM devient un facteur limitant. Dans Axis, parce-que l'aperçu du parcours d'outil est affiché par défaut, le rafraîchissement de l'écran devient une limite pratique à la taille des fichiers. Le tracé du parcours d'outil peut être désactivé dans Axis pour accélérer le chargement des fichiers conséquents. L'aperçu peut être désactivé en passant un commentaire spécial.

## 14.30 Ordre d'exécution

L'ordre d'exécution des éléments d'une ligne est défini, non pas par sa position dans la ligne mais par la liste suivante:
- Commandes O-code, optionnellement suivies par un commentaire mais aucun autre mot n'est permis sur la même ligne.
- Commentaire (message inclus).
- Positionnement du mode de vitesses (G93, G94).
- Réglage de la vitesse travail (F).
- Réglage de la vitesse de rotation de la broche (S).
- Sélection de l'outil (T).
- pin I/O de HAL (M62 à M68).
- Appel d'outil (M6).
- Marche/Arrêt broche (M3, M4, M5).
- Enregistrer l'état (M70, M73), restaurer l'état (M72), invalider l'état (M71).
- Marche/Arrêt arrosages (M7, M8, M9).
- Activation/Inhibition des correcteurs de vitesse (M48, M49, M50, M51, M52, M53).
- Commandes définies par l'opérateur (M100 à M199).
- Temporisation (G4).
- Choix du plan de travail (G17, G18, G19).
- Choix des unités de longueur (G20, G21).
- Activation/Désactivation de la compensation de rayon d'outil (G40, G41, G42)
- Activation/Désactivation de la compensation de longueur d'outil (G43, G49)

- Sélection du système de coordonnées (G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3).
- Réglage du mode de trajectoire (G61, G61.1, G64)
- Réglage du mode de déplacement (G90, G91).
- Réglage du mode de retrait (G98, G99).
- Prise d'origine (G28, G30) ou établissement du système de coordonnées (G10) ou encore, réglage des décalages d'axes (G92, G92.1, G92.2, G94).
- Effectuer un mouvement (G0 à G3, G33, G80 à G89), tel que modifié (éventuellement) par G53.
- Arrêt (M0, M1, M2, M30, M60).

## 14.31 G-Code: Bonnes pratiques

### 14.31.1 Utiliser un nombre de décimales approprié

Utiliser au plus 3 chiffres après la virgule pour l'usinage en millimètres et au plus 4 chiffres après la virgule pour l'usinage en pouces. En particulier, les contrôles de tolérance des arcs sont faits pour .001 et .0001 selon les unités actives.

### 14.31.2 Utiliser les espaces de façon cohérente

Le G-code est plus lisible quand au moins un espace apparaît avant les mots. S'il est permis d'insérer des espaces blancs au milieu des chiffres, il faut éviter de le faire.

### 14.31.3 Préférer le *format centre* pour les arcs

Les arcs en format centre (qui utilisent *I- J- K-* au lieu de *R-* ) se comportent de façon plus précise que ceux en format rayon, particulièrement pour des angles proche de 180 et 360 degrés.

### 14.31.4 Placer les codes modaux importants au début des programmes

Lorsque l'exécution correcte de votre programme dépend de paramètres modaux, n'oubliez pas de les mettre au début du programme. Des modes incorrects peuvent provenir d'un programme précédent ou depuis des entrées manuelles.

Une bonne mesure préventive consiste à placer la ligne suivante au début de tous les programmes:

```
G17 G21 G40 G49 G54 G80 G90 G94
```

(plan XY, mode mm, annulation de la compensation de rayon, et de longueur, système de coordonnées numéro 1, arrêt des mouvements, déplacements absolus, mode vitesse/minute)

Peut-être que le code modal le plus important est le réglage des unités machine. Si les codes G20 ou G21, ne sont pas inclus, selon les machines l'échelle d'usinage sera différente. D'autres valeurs comme le plan de retrait des cycles de perçage peuvent être importantes.

### 14.31.5 Ne pas mettre trop de choses sur une ligne

Ignorer le contenu de la section ordre d'exécution et ne pas écrire de ligne de code qui laisse la moindre ambiguïté.

### 14.31.6 Ne pas régler et utiliser un paramètre sur la même ligne

Ne pas *utiliser* et *définir* un paramètre sur la même ligne, même si la sémantique est bien définie. Mettre à jour une variable, à une nouvelle valeur, telle que *#1 = [#1 + #2]* est autorisé.

### 14.31.7 Ne pas utiliser les numéros de ligne

Les numéros de ligne n'apportent rien. Quand des numéros de ligne sont rapportés dans les messages d'erreur, ces numéros font référence aux numéros de lignes à l'intérieur du programme, pas aux valeurs des mots N.

### 14.31.8 Lorsque plusieurs systèmes de coordonnées sont déplacés

envisager le mode vitesse inverse du temps.

Parce que la signification d'un mot *F* en mètres par minute varie selon les axes à déplacer et parce que la quantité de matière enlevée ne dépend pas que de la vitesse travail, il peut être plus simple d'utiliser G93, vitesse inverse du temps, pour atteindre l'enlèvement de matière souhaité.

## 14.32 Axes rotatifs et linéaires

La signification du mot *F*-, exprimé en vitesse par minute, étant différente selon l'axe concerné par la commande de déplacement et parce-que la quantité de matière enlevée ne dépend pas seulement de la vitesse d'avance, il est facile d'utiliser le mode inverse du temps *G93* pour atteindre la quantité de matériaux à enlever, souhaitée.

## 14.33 Messages d'erreur courants

- *G code hors d'étendue* - Un G-code supérieur à G99 a été utilisé. L'étendue des G-codes dans LinuxCNC est comprise entre 0 et 99. Toutefois, les valeurs entre 0 et 99 ne sont pas toutes celle d'un G-code valide.
- *Utilisation d'un G code inconnu* - Un G-code à été utilisé qui n'appartient pas aux langage G-code de LinuxCNC.
- *Mot i, j, k sans Gx l'utilisant* - Les mots i, j et k doivent être utilisés sur la même ligne que leur G-code.
- *Impossible d'employer des valeurs d'axe sans G code pour les utiliser* - Les valeurs d'axe ne peuvent pas être utilisées sur une ligne sans qu'un G-code ne se trouve sur la même ligne ou qu'un G-code modal soit actif.
- *Le fichier se termine sans signe pourcent ni fin de programme* - Tout fichier G-code doit se terminer par un M2, un M30 ou être encadré par le signe %.

# Chapitre 15

# Tout le G-code de LinuxCNC

## 15.1 Conventions d'écriture du G-code

Dans une commande type, le tiret (-) signifie une valeur réelle et les signes (<>) indiquent un item facultatif.

Si *L-* est écrit dans une commande, le signe - fera référence à *Lnombre*. De la même manière, le signe - dans *H-* peut être appelé le *Hnombre* et ainsi de suite pour les autres lettres. Une valeur facultative sera écrite <*L-*>.

Dans les blocs de G-code, le mot *axes* signifie n'importe quel axe défini dans la configuration.

Une valeur réelle peut être:
– - un nombre explicite, *4* par exemple.
– - une expression, *[2+2]* par exemple.
– - une valeur de paramètre, *#88* par exemple.
– - une fonction unaire de la valeur, *acos[0]* par exemple.
Dans la plupart des cas, si des mots d'axes sont donnés parmi *XYZABCUVW*, ils spécifient le point de destination.

Les axes sont donnés dans le système de coordonnées courant, à moins qu'explicitement décrit comme étant dans le système de coordonnées absolues (machine).

Les axes sont facultatifs, tout axe omis gardera sa valeur courante.

Tout item dans un bloc de G-code, non explicitement décrit comme facultatif, sera requis. Une erreur sera signalée si un item requis est omis.

Dans les commandes, les valeurs suivant les lettres sont souvent données comme des nombres explicites. Sauf indication contraire, les nombres explicites peuvent être des valeurs réelles. Par exemple, *G10 L2* pourrait aussi bien être écrite *G[2*5] L[1+1]*. Si la valeur du paramètre *100* étaient *2*, *G10 L#100* signifierait également la même chose.

## 15.2 Table d'index du G-code

| Sections | Descriptions |
|---|---|
| G0 | Interpolation linéaire en vitesse rapide |
| G1 | Interpolation linéaire en vitesse travail |
| G2/G3 | Interpolation circulaire sens horaire/anti-horaire |
| G4 | Temporisation |
| G5 | Spline cubique |
| G5.1 | B-Spline quadratique |
| G5.2 | NURBS, ajout point de contrôle |
| G5.3 | NURBS, exécute |
| G7 | Mode diamètre (sur les tours) |
| G8 | Mode rayon (sur les tours) |

| Sections | Descriptions |
|---|---|
| G10 L1 | Ajuste les valeurs de l'outil en table d'outils |
| G10 L10 | Modifie les valeurs de l'outil dans la table d'outils |
| G10 L11 | Fixe les valeurs de l'outil dans la table d'outils |
| G10 L2 | Fixe l'origine d'un système de coordonnées |
| G10 L20 | Fixe l'origine du système de coord. aux valeurs calculées |
| G18 G19 | Choix du plan de travail |
| G20 G21 | Unités machine |
| G28 G28.1 | Aller à une position prédéfinie |
| G30 G30.1 | Aller à une position prédéfinie |
| G33 | Mouvement avec broche synchronisée |
| G33.1 | Taraudage rigide |
| G38 | Mesures au palpeur |
| G40 | Révocation de la compensation de rayon d'outil |
| G41 G42 | Compensation de rayon d'outil |
| G41.1 G42.1 | Comp. dynamique de rayon d'outil à gauche/à droite |
| G43 | Compensation de longueur d'outil d'après une table d'outils |
| G43.1 | Compensation dynamique de longueur d'outil |
| G49 | Révocation de la compensation de longueur d'outil |
| G53 | Déplacements en coordonnées machine (Absolues) |
| G54 à G59.3 | Choix du système de coordonnées (1 à 9) |
| G61 G61.1 | Mode trajectoire exacte/mode arrêts exacts |
| G64 | Mode trajectoire continue avec tolérance |
| G73 | Cycle de perçage avec brise copeau |
| G76 | Cycle de filetage multipasses (tour) |
| G80 | Révocation des codes modaux |
| G81 | Cycle de perçage |
| G82 | Autres cycles de perçage |
| G83 | Perçage avec débourrage |
| G84 | Taraudage à droite *(pas encore implémenté)* |
| G85 | Alésage, retrait en vitesse travail |
| G86 | Alésage, retrait en vitesse rapide |
| G87 | Cycle d'alésage arrière *(pas encore implémenté)* |
| G88 | Cycle alésage, Stop, Retrait manuel *(pas encore implémenté)* |
| G89 | Cycle d'alésage avec tempo, recul vitesse travail |
| G90 | Types de déplacement |
| G90.1 G91.1 | Arc I,J,K, centre absolu ou relatif |
| G92 | Décalages d'origines avec mise à jour des paramètres |
| G92.1 G92.2 | Révocation des décalages d'origine |
| G92.3 | Applique contenu des paramètres aux déc. d'origine |
| G93 | Modes de vitesse |
| G96 | Vitesse de coupe constante (IPM ou m/mn) |
| G97 | Vitesse en tours par minute |
| G98 | Options de retrait des cycles de perçage |

## 15.3 G0 Interpolation linéaire en vitesse rapide

```
G0 axes
```

Pour un mouvement linéaire en vitesse rapide, programmer *G0 axes*, tous les mots d'axe sont facultatifs. Le *G0* est facultatif si le mode mouvement courant est déjà *G0*. Cela produit un mouvement linéaire vers le point de destination à la vitesse rapide courante (ou moins vite si la machine n'atteint pas cette vitesse). Il n'est pas prévu d'usiner la matière quand une commande G0 est exécutée. Un G0 seul peut être utilisé pour passer le mode de mouvement courant en G0.

**Exemple avec G0:**

```
G90 (Fixe les édplacements en mode absolu)
G0 X1 Y-2.3 (mouvement élinaire en vitesse rapide du point courant à X1 Y-2.3)
M2 (fin de programme)
```

– Voir les sections G90 et M2 pour plus d'informations.

Si la compensation d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section sur la compensation de d'outil.

Si *G53* est programmé sur la même ligne, le mouvement sera également différent, voir la section sur les mouvements en coordonnées absolues.

C'est une erreur si:
– Un mot d'axe est indiqué sans valeur réelle.
– Un mot d'axe est indiqué qui n'est pas configuré.

## 15.4 G1 Interpolation linéaire en vitesse travail

```
G1 axes
```

Pour un mouvement linéaire en vitesse travail, (pour usiner ou non) programmer *G1 axes*, tous les mots d'axe sont facultatifs. Le *G1* est facultatif si le mode de mouvement courant est déjà *G1*. Cela produira un mouvement linéaire vers le point de destination à la vitesse de travail courante (ou moins vite si la machine n'atteint pas cette vitesse). Un G1 seul peut être utilisé pour passer le mode de mouvement courant en G1.

**Exemple avec G1:**

```
G90 (Fixe les édplacements en mode absolu)
G1 X1.2 Y-3 F10 (mouvement élinaire à 10 éunits/mn du point courant à X1.2 Y-3)
Z-2.3 (mouvement élinaire à 10 éunits/mn du point courant à Z-2.3)
Z1 F25 (mouvement élinaire de l'axe Z à 25 éunits/mn vers Z1)
M2 (Fin de programme)
```

– Voir les sections G90 et M2 pour plus d'informations.

Si la compensation d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section sur la compensation d'outil. Si *G53* est programmé sur la même ligne, le mouvement sera également différent, voir la section sur les mouvements en coordonnées absolues.

C'est une erreur si:
– - Aucune vitesse d'avance travail n'est fixée.
– - un mot d'axe est indiqué sans valeur réelle.
– - un mot d'axe est indiqué qui n'est pas configuré.

## 15.5 G2, G3 Interpolation circulaire en vitesse travail

```
G2 ou G3 axes édcalages (format centre)
G2 ou G3 axes R- (format rayon)
G2 ou G3 édcalages <P-> (cercles complet)
```

Un mouvement circulaire ou hélicoïdal est spécifié en sens horaire avec *G2* ou en sens anti-horaire avec *G3*. La direction est vue depuis le côté positif de l'axe autour duquel le mouvement se produit.

Les axes de cercle ou les hélicoïdes, doivent être parallèles aux axes X, Y ou Z du système de coordonnées machine. Les axes (ou, leurs équivalents, les plans perpendiculaires aux axes) sont sélectionnés avec *G17* (axe Z, plan XY), *G18* (axe Y, plan XZ), ou *G19* (axe X, plan YZ). Les plans *17,1*, *18,1* et *19,1* ne sont pas actuellement pris en charge. Si l'arc est circulaire, il se trouve dans un plan parallèle au plan sélectionné.

Pour programmer un hélicoïde, inclure le mot d'axe perpendiculaire au plan de l'arc. Par exemple, si nous sommes dans le plan *G17*, inclure un mot *Z*, ceci provoquera un mouvement de l'axe *Z* vers valeur programmée durant tout le mouvement circulaire *XY*.

Pour programmer un arc supérieur à un tour complet, utiliser un mot *P* spécifiant alors le nombre de tours complets en plus de l'arc. Si *P* n'est pas spécifié, le comportement sera comme si *P1* avait été donné: ceci étant, un seul tour complet ou partiel sera effectué, donnant un arc plus petit ou égal à un tour complet. Par exemple, si un arc de 180° est programmé avec P2, le mouvement résultant sera d'un tour et demi. Pour chaque incrément de P au delà de 1, un tour complet sera ajouté à l'arc programmé. Les arcs hélicoïdaux multitours sont supportés ce qui donne des mouvements très intéressants pour usiner des alésages ou des filetages.

Si une ligne de G-code crée un arc et inclus le mouvement d'un axe rotatif, l'axe rotatif tournera à vitesse constante de sorte que le mouvement de l'axe rotatif commence et se termine en même temps que les autres axes XYZ. De telles lignes sont rarement programmées.

Si la compensation d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir les sections sur G40 et sur G41-G42.

Le centre de l'arc est absolu ou relatif, tel que fixé par G90.1 ou G91.1, respectivement.

C'est une erreur si:
– Aucune vitesse d'avance travail n'est spécifiée.
Deux formats sont possibles pour spécifier un arc: Le format centre et le format rayon.

### 15.5.1 Arc au format centre (format recommandé)

Les arcs au format centre sont plus précis que les arcs au format rayon, c'est le format à privilégier.

La distance entre la position courante et le centre de l'arc et, facultativement, le nombre de tours, sont utilisés pour programmer des arcs inférieurs au cercle complet. Il est permis d'avoir le point final de l'arc égal à la position courante.

Le décalage entre le centre de l'arc et la position courante ainsi que facultativement, le nombre de tours, sont utilisés pour programmer des cercles complets.

Une erreur d'arrondi peut se produire quand un arc est programmé avec une précision inférieure à 4 décimales (0.0000) pour les pouces et à moins de 3 décimales (0.000) pour les millimètres.

**Arc en mode distance relative** Les décalages par rapport au centre de l'arc sont des distances relatives au point de départ de l'arc. Le mode distance relative de l'arc est le mode par défaut.

Un ou plusieurs mots d'axe et un ou plusieurs décalages doivent être programmés pour un arc qui fait moins de 360 degrés.

Aucun mot d'axe mais un ou plusieurs décalages doivent être programmés pour un cercle complet. Le mot *P*, par défaut à 1, est facultatif.

Pour d'avantage d'information sur les arcs en mode relatif, voir la section G91.1.

**Arc en mode distance absolue** Les décalages par rapport au centre de l'arc sont des distances absolues depuis la position 0 courante des axes (origine machine).

Un ou plusieurs mots d'axe et *tous* les décalages doivent être programmés pour les arcs de moins de 360 degrés.

Aucun mots d'axe mais tous les décalages doivent être programmés pour un cercle complet. Le mot *P*, par défaut à 1, est facultatif.

Pour d'avantage d'information sur les arcs en mode absolu, voir la section G90.1.

**Plan XY (G17)**

```
G2 ou G3 <X- Y- Z- I- J- P->
```

– *Z* - hélicoïde
– *I* - décalage en X
– *J* - décalage en Y
– *P* - nombre de tours

**Plan XZ (G18)**

```
G2 ou G3 <X- Z- Y- I- K- P->
```

– *Y* - hélicoïde
– *I* - décalage en X
– *K* - décalage en Z
– *P* - nombre de tours

**YZ-plane (G19)**

```
G2 ou G3 <Y- Z- X- J- K- P->
```

– *X* - hélicoïde
– *J* - décalage en Y
– *K* - décalage en Z
– *P* - nombre de tours

C'est une erreur si:
– Aucune vitesse d'avance travail n'est fixée avec le mot F.
– Aucun décalage n'est programmé.
– Quand l'arc est projeté dans le plan courant, la distance depuis le point courant et le centre diffère de la distance entre le point final et le centre, de plus de (.05 pouce/.5 mm) OU ((.0005 pouce/.005mm) ET .1% du rayon).

Déchiffrer le message d'erreur *Le rayon à la fin de l'arc diffère de celui du début:*
– *début* - position courante
– *centre* - la position du centre telle que calculée avec les paramètres I,J ou K
– *fin* - le point final programmé
– *r1* - le rayon entre le point de départ et le centre
– *r2* - le rayon entre le point final et le centre

### 15.5.2 Exemples d'arcs au format centre

Calculer des arcs à la main peut être difficile. Il est possible de dessiner l'arc à l'aide d'un programme de DAO pour obtenir les coordonnées et les décalages. Garder à l'esprit les tolérances, il pourrait être nécessaire de modifier la précision de la DAO pour obtenir les résultats souhaités. Une autre option consiste à calculer les coordonnées et les décalages en utilisant des formules. Comme vous pouvez le voir sur la figure suivante un triangle peut être formé à partir de la position courante, de la position de fin et du centre de l'arc.

Sur la figure suivante, vous voyez que la position de départ est X0 Y0, la position finale est X1 Y1. La position du centre de l'arc est X1 Y0. Ceci donne un décalage de 1 depuis la position de départ sur l'axe X et 0 sur l'axe Y. Dans ce cas seul le décalage I est nécessaire.

Le G-code de cet exemple serait:

```
G0 X0 Y0
G2 X1 Y1 I1 F10 (arc en sens horaire dans le plan XY)
```

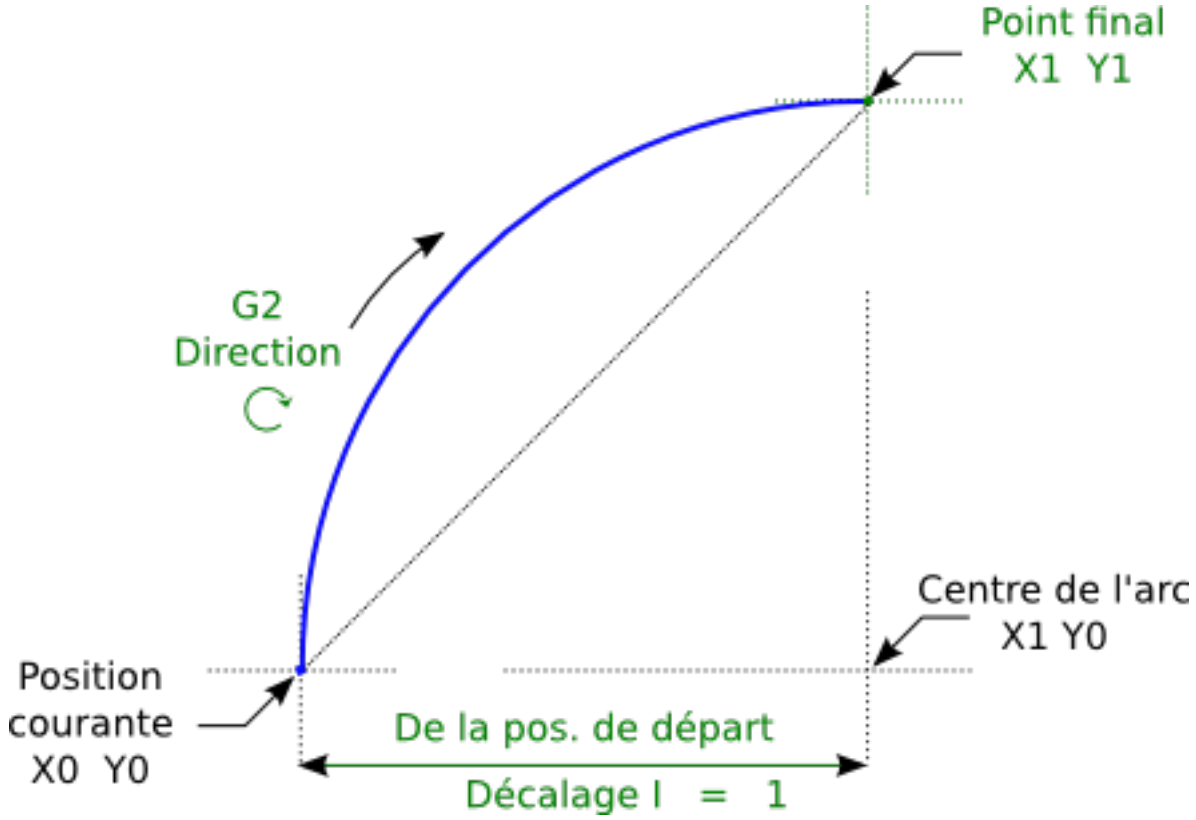



FIGURE 15.1 – Exemple avec G2

Dans cet autre exemple, nous pouvons voir les différences de décalages pour Y selon que nous faisons un mouvement G2 ou un mouvement G3. Pour le mouvement G2 la position de départ est en X0 Y0, alors que pour le mouvement G3 elle est en X0 Y1. Le centre de l'arc est en X1 Y0.5 pour les deux. Le décalage J du mouvement G2 est 0.5 alors que celui du mouvement G3 est -0.5.

Le G-code de cet exemple serait:

```
G0 X0 Y0
G2 X0 Y1 I1 J0.5 F25 (arc en sens horaire dans le plan XY)
G3 X0 Y0 I1 J-0.5 F25 (arc en sens anti-horaire dans le plan XY)
```

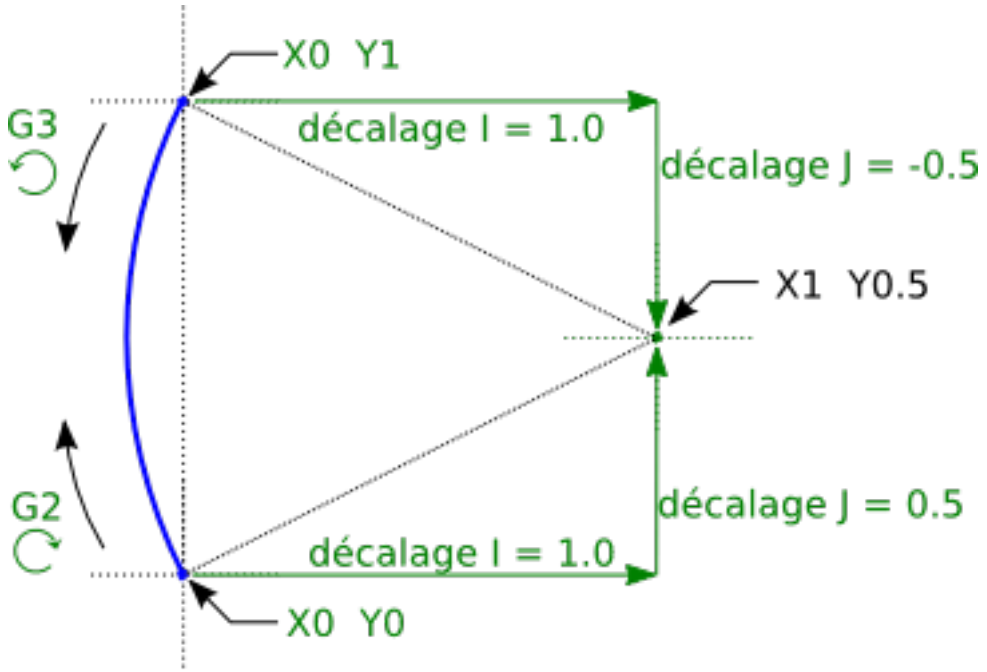



FIGURE 15.2 – Exemple avec G2-G3

Voici un exemple au format centre pour usiner une hélice:

```
G0 X0 Y0 Z0
G17 G2 X10 Y16 I3 J4 Z-1 (Arc éïhlicodal avec ajout de Z)
```

**exemple avec P**

```
G0 X0 Y0 Z0
G2 X0 Y1 Z-1 I1 J0.5 P2 F25
```

Cet exemple signifie, faire un mouvement circulaire ou hélicoïdal en sens horaire (vu du côté positif sur l'axe Z), dont l'axe est parallèle à l'axe Z, se terminant en X10, Y16 et Z9, avec son centre décalé de 3 unités dans la direction X, par rapport à la position X courante. Son centre décalé dans la direction Y de 4 unités depuis la position Y courante. Si la position courante est X7, Y7 au départ, le centre sera en X10, Y11. Si la valeur de départ en Z est 9, ce sera un arc circulaire. Autrement, ce sera un arc hélicoïdal. Le rayon de cet arc serait de 5 unités.

Dans le format centre, le rayon de l'arc n'est pas spécifié, mais il peut facilement être trouvé puisque c'est la distance entre le point courant et le centre du cercle, ou le point final de l'arc et le centre.

### 15.5.3 Arcs au format rayon (format non recommandé)

```
G2 ou G3 axes R-
```

– R - rayon depuis la position courante

Ce n'est pas une bonne pratique de programmer au format rayon des arcs qui sont presque des cercles entiers ou des demi-cercles, car un changement minime dans l'emplacement du point d'arrivée va produire un changement beaucoup plus grand dans l'emplacement du centre du cercle (et donc, du milieu de l'arc). L'effet de grossissement est tellement important, qu'une erreur d'arrondi peut facilement produire un usinage hors tolérance. Par exemple, 1% de déplacement de l'extrémité d'un arc de 180 degrés produit 7% de déplacement du point situé à 90 degrés le long de l'arc. Les cercles presque complets sont encore pires. Autrement, l'usinage d'arcs, inférieurs à 165 degrés ou compris entre 195 et 345 degrés sera possible.

Dans le format rayon, les coordonnées du point final de l'arc, dans le plan choisi, sont spécifiées en même temps que le rayon de l'arc. Programmer *G2 axes R-* (ou utiliser *G3* au lieu de *G2* ). R est le rayon. Les mots d'axes sont facultatifs sauf au moins un des deux du plan choisi, qui doit être utilisé. Un rayon positif indique que l'arc fait moins de 180 degrés, alors qu'un rayon négatif indique un arc supérieur à 180 degrés. Si l'arc est hélicoïdal, la valeur du point d'arrivée de l'arc dans les coordonnées de l'axe perpendiculaire au plan choisi sera également spécifiée.

C'est une erreur si:
– Les deux mots d'axes pour le plan choisi sont omis.
– Le point d'arrivée de l'arc est identique au point courant.
Voici un exemple de commande pour usiner un arc au format rayon:

```
G17 G2 X10 Y15 R20 Z5 (arc au format rayon)
```

Cet exemple signifie, faire un mouvement en arc ou hélicoïdal en sens horaire (vu du côté positif de l'axe Z), se terminant en X=10, Y=15 et Z=5, avec un rayon de 20. Si la valeur de départ de Z est 5, ce sera un arc de cercle parallèle au plan XY sinon, ce sera un arc hélicoïdal.

## 15.6 G4 Tempo

```
G4 P-
```

– *P* - durée de la temporisation en secondes (un flottant)

Les axes s'immobiliseront pour une durée de P secondes. Cette commande n'affecte pas la broche, les arrosages ni les entrées/-sorties.

C'est une erreur si:
– Le nombre P est négatif ou n'est pas spécifié.

## 15.7 G5 Spline cubique

```
G5 X- Y- <I- J-> P- Q-
```

– *I* - offset incrémental en X, du point de départ au premier point de contrôle
– *J* - offset incrémental en Y, du point de départ au premier point de contrôle
– *P* - offset incrémental en X, du point de départ au second point de contrôle
– *Q* - offset incrémental en Y, du point de départ au second point de contrôle

G5 crée une B-spline cubique dans le plan XY avec les axes X et Y seuls. P et Q doivent être tous les deux spécifiés pour chaque commande G5.

Pour la première d'une série de commandes G5, I et J doivent être tous les deux spécifiés. Pour les commandes G5 suivantes de la série, soit I et J sont spécifiés tous les deux, soit aucun ne l'est. Si aucun n'est spécifié, la direction de départ de ce cube rejoindra automatiquement la direction de fin du cube précédent (comme si I et J étaient les négatifs des P et Q précédents).

Par exemple, pour programmer une courbe en forme de N:

**G5 Simple spline cubique initiale**

```
G90 G17
G0 X0 Y0
G5 I0 J3 P0 Q-3 X1 Y1
```

Une seconde courbe en N qui s'attache doucement à celle-ci peux maintenant être faite sans spécifier I et J:

**G5 Simple spline cubique subséquente**

```
G5 P0 Q-3 X2 Y2
```

C'est une erreur si:
– P et Q ne sont pas spécifiés tous les deux
– Un seul, de I ou J est spécifié
– Aucun de I ou J n'est spécifié à la première série de commandes G5
– Un axe autre que X ou Y est spécifié
– Le plan courant n'est pas G17

## 15.8 G5.1 Spline quadratique

```
G5.1 X- Y- I- J-
```

– *I* - Offset incrémental en X, du point de départ au point de contrôle
– *J* - Offset incrémental en Y, du point de départ au point de contrôle

G5.1 crée une B-spline quadratique dans le plan XY avec les seuls axes X et Y. Ne pas spécifier I ou J donne un offset nul pour l'axe non spécifié, un ou les deux doivent donc être donnés.

Par exemple, pour programmer une parabole, entre l'origine X-2 Y4 et X2 Y4:

**G5.1 Simple spline quadratique**

```
G90 G17
G0 X-2 Y4
G5.1 X2 I2 J-8
```

C'est une erreur si:
– Les offsets I et J ne sont pas spécifiés ou sont à zéro
– Un autre axe que X ou Y est spécifié
– Le plan actif n'est pas G17

## 15.9 G5.2 G5.3 Block NURBS

```
G5.2 <P-> <X- Y-> <L->
X- Y- <P->
...
G5.3
```

**AVERTISSEMENT**
G5.2, G5.3 sont expérimentaux, il n'ont pas encore été testés totalement.

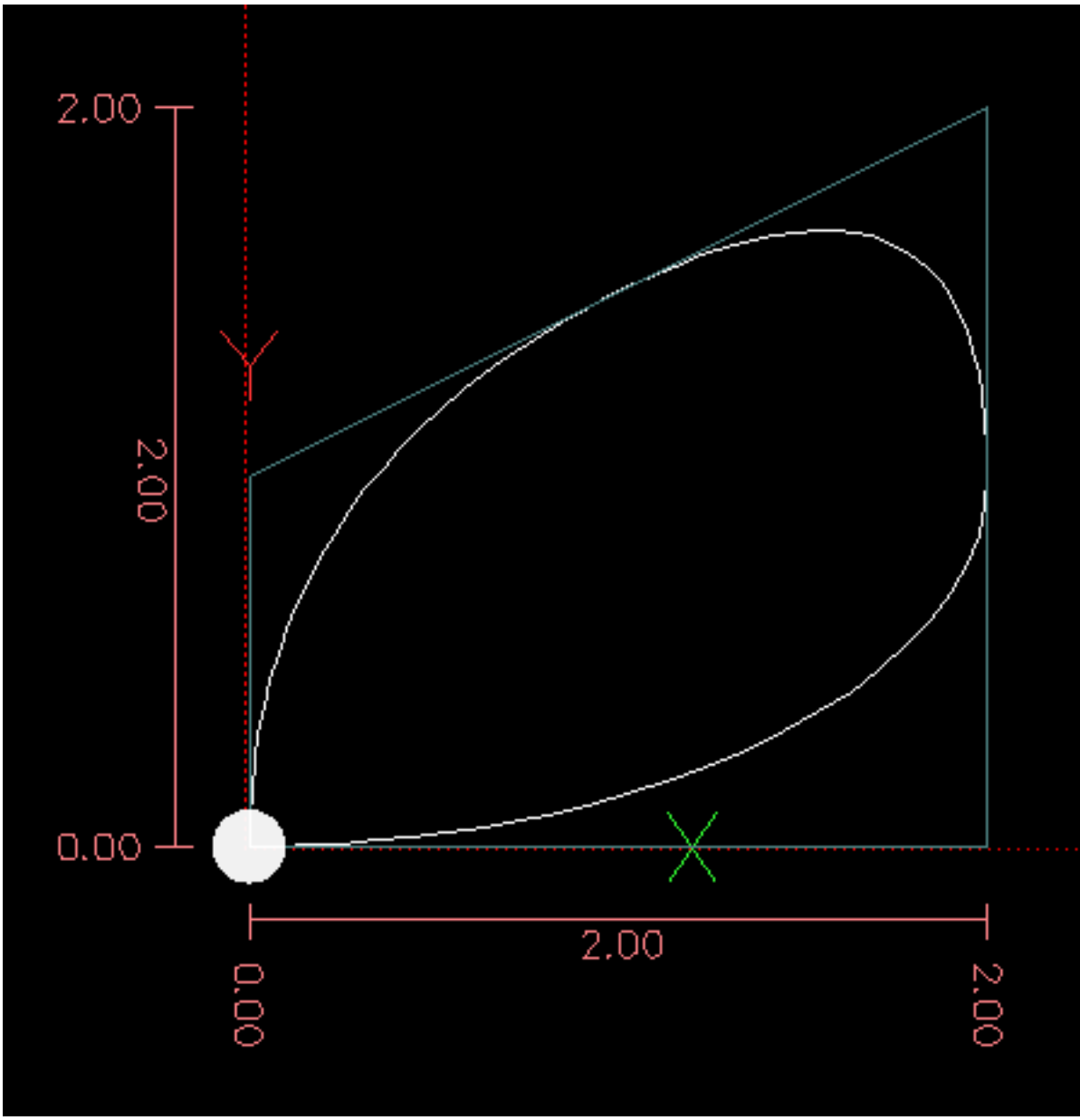
G5.2 est pour ouvrir un bloc de données définissant un NURBS et G5.3 pour fermer le bloc de données. Dans les lignes entre ces deux codes, les points de contrôle de la courbe sont définis avec deux éléments, leur *poids* relatif (P) et le paramètre (L) qui détermine l'ordre de la courbe.

Les coordonnées courantes, avant la premiére commande G5.2, est toujours prise comme premier point de contrôle du NURBS. Pour définir le poids pour le premier point de contrôle, premièrement programmer G5.2 P- sans donner X ni Y.

Le poids par défaut si P n'est pas spécifié est 1. L'ordre par défaut si L n'est pas spécifié est 3.

**G5.2 Exemple**

```
G0 X0 Y0 (mouvement en vitesse rapide)
F10 (set feed rate)
G5.2 P1 L3
     X0 Y1 P1
     X2 Y2 P1
     X2 Y0 P1
     X0 Y0 P2
G5.3
; Les mouvements en vitesse rapide montrent le êmme parcours sans le bloc NURBS
G0 X0 Y1
   X2 Y2
   X2 Y0
   X0 Y0
M2
```




**Simple sortie NURBS**

D'autres informations sur NURBS sont disponibles ici:

http://wiki.linuxcnc.org/cgi-bin/wiki.pl?NURBS

## 15.10 G7 Mode diamètre sur les tours

```
G7
```

Sur un tour, programmer *G7* pour passer l'axe X en mode diamètre. En mode diamètre, les mouvements de l'axe X font la moitié de la cote programmée. Par exemple, X10 placera l'outil à 5 unités du centre, ce qui produira bien une pièce d'un diamètre de 10 unités.

## 15.11 G8 Mode rayon sur les tours

```
G8
```

Sur un tour, programmer *G8* pour passer l'axe X en mode rayon. En mode rayon, les mouvements de l'axe X sont égaux à la cote programmée. Ce qui signifie que X10 placera l'outil à 10 unités du centre et aura pour résultat une pièce d'un diamètre de 20 unités. G8 est le mode par défaut à la mise sous tension.

## 15.12 G10 L1 Ajustements dans la table d'outils

```
G10 L1 P- axes <R- I- J- Q->
```

– *P* - numéro d'outil
– *R* - rayon de bec
– *I* - angle frontal (tour)
– *J* - angle arrière (tour)
– *Q* - orientation (tour)

*G10 L1* ajuste les valeurs de la table d'outils pour l'outil N°*P* aux valeurs passées dans les paramètres. Les nouvelles valeurs peuvent être passées depuis un programme ou depuis la fenêtre d'entrées manuelles (MDI). Un G10 L1 valide, réécrit et recharge la table d'outils.

**Exemples avec G10 L1:**

```
G10 L1 P1 Z1.5 (fixe le édcalage en Z de l'outil 1 à 1.5 de l'origine machine)
G10 L1 P2 R0.15 Q3 (fixe le rayon de bec de l'outil 2 à 0.15 avec une orientation 3)
```

C'est une erreur si:
– La compensation d'outil est active
– Le mot P n'est pas spécifié
– Le mot P ne correspond pas à un numéro d'outil valide de la table d'outils.
D'autres informations sur l'orientation des outils de tour sont disponibles ici.

## 15.13 G10 L2 Établissement de l'origine d'un système de coordonnées

```
G10 L2 P- <axes R->
```

– *P* - système de coordonnées (0 à 9)
– *R* - rotation autour de l'axe Z

G10 L2 décale l'origine des axes dans le système de coordonnées spécifié par la valeur du mot d'axe. Le décalage s'effectue à partir de l'origine machine établie par la prise d'origine machine (homing). Les valeurs de ce décalage vont remplacer toutes celles en effet sur le système de coordonnées spécifié. Les mots d'axe inutilisés resteront inchangés.

Programmer P0 à P9 pour spécifier le système de coordonnées à décaler.

TABLE 15.1: Systèmes de coordonnées

| Valeur P | Système de coordonnées | G-code |
|---|---|---|
| 0 | Actif courant | n/a |
| 1 | 1 | G54 |
| 2 | 2 | G55 |
| 3 | 3 | G56 |
| 4 | 4 | G57 |
| 5 | 5 | G58 |
| 6 | 6 | G59 |
| 7 | 7 | G59.1 |
| 8 | 8 | G59.2 |
| 9 | 9 | G59.3 |

Facultativement, programmer *R* pour indiquer la rotation des axes *XY* autour de l'axe *Z*. La direction de rotation est anti-horaire comme vue depuis le côté positif de l'axe Z.

Tous les mots d'axe sont facultatifs.

Être en mode relatif (*G91*) est sans effet sur *G10 L2*.

Concepts importants:
- G10 L2 Pn ne change pas l'actuel système de coordonnées par celui spécifié par P, il est nécessaire d'utiliser G54 à 59.3 pour sélectionner le système de coordonnées.
- Quand un mouvement de rotation est en cours, jogger un axe, déplacera celui-ci seulement dans le sens négatif ou positif et non pas le long de l'axe de rotation.
- Si un décalage d'origine créé avec *G92* est actif avant la commande *G10 L2*, il reste actif après.
- Le système de coordonnées dont l'origine est définie par la commande *G10* peut être actif ou non au moment de l'exécution de *G10*. Si il est actif à ce moment là, les nouvelles coordonnées prennent effet immédiatement.

C'est une erreur si:
- Le nombre P n'est pas évalué comme étant un nombre entier compris entre 0 et 9.
- Un axe est programmé mais n'est pas défini dans la configuration.

**Premier exemple avec G10 L2:**

```
G10 L2 P1 X3.5 Y17.2
```

Place l'origine du premier système de coordonnées (celui sélectionné par G54) au points X3.5 et Y17.2 (en coordonnées absolues). La coordonnée Z de l'origine, ainsi que les coordonnées de tous les autres axes, restent inchangées puisque seuls X et Y étaient spécifiés.

**Deuxième exemple avec G10 L2:**

```
G10 L2 P1 X0 Y0 Z0 (érvoque les édcalages en X, Y et Z du èsystme °N1)
```

L'exemple précédent fixe les origines XYZ du système de coordonnées G54, à l'origine machine.

Les systèmes de coordonnées sont décrits en détail ici.

## 15.14 G10 L10 modifie les offsets d'outil dans la table d'outils

```
G10 L10 P- axes <R- I- J- Q->
```

- *P* - numéro d'outil
- *R* - rotation autour de l'axe Z
- *I* - angle frontal (tour)

– *J* - angle arrière (tour)
– *Q* - orientation (tour)

G10 L10 modifie les valeurs de l'outil *P* dans la table d'outils, de sorte que si la compensation d'outil est rechargée, avec la machine à la position courante et avec les G5x et G92 actifs, les coordonnées courantes pour l'axe spécifié deviendront les coordonnées spécifiées. Les axes non spécifiés dans la commande G10 L10 ne seront pas modifiés.

**Exemple avec G10 L10:**

```
M6 T1 G43 (appel l'outil 1 et active la correction de longueur d'outil)
G10 L10 P1 Z1.5 (fixe la position courante en Z à 1.5 dans la table d'outils)
G43 (recharge l'offset de longueur d'outil depuis la table d'outils émodifie)
M2 (fin de programme)
```

Pour d'autres détals voir les commandes M6, Tn et G43/G43.1.

C'est une erreur si:
– La compensation d'outil est activée.
– Le mot P n'est pas spécifié.
– Le mot P ne correspond pas à un numéro d'outil valide de la table d'outils.

## 15.15 G10 L11 modifie les offsets d'outil dans la table d'outils

```
G10 L11 P- axes <R- I- J- Q->
```

– *P* - numéro d'outil
– *R* - rotation autour de l'axe Z
– *I* - angle frontal (tour)
– *J* - angle arrière (tour)
– *Q* - orientation (tour)

G10 L11 est identique à G10 L10 excepté qu'au lieux de fixer les valeurs par rapport aux décalages de coordonnées courants, il les fixe de sorte que les coordonnées courantes deviennent celles spécifiées par les paramètres si la nouvelle compensation d'outil est rechargée et que la machine est placée dans le système de coordonnées G59.3, système sans aucun décalage G92 actif.

Ceci permet à l'utilisateur de fixer le système de coordonnées G59.3 à un point fixe de la machine et d'utiliser cet emplacement pour mesurer l'outil sans s'occuper des autres décalages courants actifs.

C'est une erreur si:
– La compensation d'outil est activée
– Le mot P n'est pas spécifié.
– Le mot P ne correspond pas à un numéro d'outil valide de la table d'outils.

## 15.16 G10 L20 Établissement de l'origine d'un système de coordonnées

```
G10 L20 P- axes
```

– *P* - système de coordonnées (0-9)

G10 L20 est similaire à G10 L2 excepté qu'au lieu d'ajuster les offsets à des valeurs données, il les place à des valeurs calculées de sorte que les coordonnées courantes deviennent les valeurs données en paramètres.

**Exemple avec G10 L20:**

```
G10 L20 P1 X1.5 (fixe la position courante en X du èsystme de écoordonnes G54à
 1.5)
```

C'est une erreur si:
– Le nombre P n'est pas évalué comme une entier compris entre 0 et 9.
– Un axe non défini dans la configuration est programmé.

## 15.17 G17 à G19.1 Choix du plan de travail

Ces codes sélectionnent le plan de travail courant comme décrit ci-dessous:
– G17 - XY (par défaut)
– G18 - ZX
– G19 - YZ
– G17.1 - UV
– G18.1 - WU
– G19.1 - VW
Les plans UV, WU et VW ne supportent pas les arcs. Il est de bonne pratique d'inclure la sélection du plan de travail dans le préambule du programme G-code. Les effets de la sélection d'un plan de travail sont discutés dans la section sur les arcs.

## 15.18 G20, G21 Choix des unités machine

– *G20* - pour utiliser le pouce comme unité de longueur.
– *G21* - pour utiliser le millimètre comme unité de longueur.

C'est toujours une bonne pratique de programmer soit *G20*, soit *G21*, dans le préambule du programme, avant tout mouvement et de ne plus en changer ailleurs dans le programme.

## 15.19 G28, G28.1 Aller à une position prédéfinie

**AVERTISSEMENT**
Pour une bonne répétabilité de la position et que la position soit correctement enregistrée avec G28.1, faire la prise d'origine générale avant d'utiliser G28.

G28 utilise les valeurs enregistrées dans les paramètres 5161 à 5166 comme points finaux des mouvements des axes X Y Z A B C U V W. Les valeurs des paramètres sont des coordonnées machine *absolues*, en unités machine natives, telles que fixées dans le fichier ini. Tous les axes définis dans le fichier ini seront déplacés lors d'un G28.
– G28 - effectue un mouvement en vitesse rapide de la position courante à la position *absolue* enregistrée dans les paramètres 5161 à 5166.
– *G28 axes* - effectue un déplacement en vitesse rapide à la position spécifiée par *axes* y compris les décalages, puis effectuera un mouvement en vitesse rapide aux coordonnées *absolues* stockées dans les paramètres 5161 à 5166 pour les axes spécifiés.
– G28.1 - enregistre la position *absolue* courante dans les paramètres 5161 à 5166.
**Exemple avec G28**

```
G28 Z2.5 (vitesse rapide vers Z2.5 puis emplacement éespcifi dans les èparamtres ↩
    éenregistrs de G28)
```

C'est une erreur si:
– La compensation d'outil est active.

## 15.20 G30, G30.1 Aller à une position prédéfinie

**AVERTISSEMENT**
Pour une bonne répétabilité de la position et que la position soit correctement enregistrée avec G30.1, faire la prise d'origine générale avant d'utiliser G30.

– *G30* - effectue un mouvement en vitesse rapide de la position courante à la position *absolue* stockée dans les paramètres 5181 à 5186. Les valeurs stockées dans les paramètres font référence au système de coordonnées absolues qui est le système de coordonnées machine.
– *G30 axes* - effectue un déplacement en vitesse rapide depuis la position courante jusqu'à la position spécifiée par *axes*, y compris les décalages, suivi d'un mouvement rapide à la position *absolue* stockée dans les paramètres 5181 à 5186 pour les axes spécifiés. Les axes non spécifiés ne bougeront pas.
– *G30.1* - enregistre la position absolue courante dans les paramètres 5181 à 5186.

**Note**
Les paramètres de *G30* peuvent être utilisés pour déplacer l'outil quand un M6 est programmé avec la variable *[TOOL_CHANGE_AT_G30]=1* dans la section *[EMCIO]* du fichier ini.

**Exemple avec G30**

```
G30 Z2.5 (mvt rapide à Z2.5 puis édplacement selon les èparamtres de G30é
stocks)
```

C'est une erreur si:
– La compensation de d'outil est active.

## 15.21 G33 Mouvement avec broche synchronisée

```
G33 X- Y- Z- K-
```

– *K* - distance par tour

Pour un mouvement avec broche synchronisée dans une direction, programmer *G33 X- Y- Z- K-* où K donne la longueur du mouvement en XYZ pour chaque tour de broche. Par exemple, si il commence à *Z=0*, *G33 Z-1 K.0625* produira un mouvement d'un pouce de long en Z en même temps que 16 tours de broche. Cette commande peut être la base d'un programme pour faire un filetage de 16 filets par pouce. Un autre exemple en métrique, *G33 Z-15 K1.5* produira un mouvement de 15mm de long pendant que la broche fera 10 tours soit un pas de 1.5mm.

Les mouvements avec broche synchronisée utilisent l'index de broche et les pins *spindle at speed* pour le filetage multi-passes. Un mouvement avec *G33* se termine au point final programmé.

**Note**
K suit la ligne d'avance décrite par *X- Y- Z-*. K n'est pas parallèle à l'axe Z si les points d'arrivée des axes X et Y sont utilisés, par exemple pour réaliser un filetage conique.

**Informations techniques** Au début de chaque passe G33, LinuxCNC utilise la vitesse de broche et les limites d'accélération de la machine pour calculer combien de temps prendra Z pour accélérer après chaque impulsion d'index et détermine de combien de degrés la broche tournera pendant ce temps là. Il ajoute alors cet angle à la position de l'index puis calcule la position de Z utilisant l'angle de broche correct. Cela signifie que Z aura atteints la position correcte juste en fin d'accélération à la bonne vitesse, il peux immédiatement usiner le bon filetage.

**Connections de hAL** Les pins *motion.spindle-at-speed* et l'index *encoder.n.phase-Z* pour la broche doivent être connectés dans le fichier HAL pour que G33 soit opérationnel. Voir le Manuel de l'intégrateur pour plus d'informations sur les mouvements synchronisés avec la broche.

**Exemple avec G33:**

```
G90 (mode distance absolue)
G0 X1 Z0.1 (positionnement en vitesse rapide)
S100 M3 (broche en rotation à 100tr/mn)
G33 Z-2 K0.125 (mouvement vers Z -2 avec une avance de 0.125 par tour)
G0 X1.25 (mouvement de édgagement en vitesse rapide)
Z0.1 (mouvement en vitesse rapide à Z0.1)
M2 (fin de programme)
```

– Voir les sections G90, G0 et M2 pour plus d'informations.

C'est une erreur si:
– Tous les axes sont omis.
– La broche ne tourne pas quand cette commande est exécutée.
– Le mouvement linéaire requis excède les limites de vitesse machine en raison de la vitesse de broche.

## 15.22 G33.1 Taraudage Rigide

```
G33.1 X- Y- Z- K-
```

– *K* - distance par tour

Pour un taraudage rigide avec broche synchronisée et mouvement de retour, programmer *G33.1 X- Y- Z- K-* où *K-* donne la longueur du mouvement pour chaque tour de broche. Un mouvement de taraudage rigide suit cette séquence:

**AVERTISSEMENT**
Si pour un taraudage rigide, les coordonnées X et Y spécifiées ne sont pas les coordonnées courantes lors de l'appel de G33.1, le mouvement ne s'effectuera pas le long de l'axe Z mais de la position courante jusqu'aux coordonnées X et Y spécifiées.

1. Un mouvement aux coordonnées spécifiées, synchronisé avec la rotation de la broche, avec le ratio donné et débutant à l'impulsion d'index du codeur de broche.
2. Quand le point final est atteint, la commande inverse le sens de rotation de la broche (ex: de 300 tours/mn en sens horaire à 300 tours/mn en sens anti-horaire)
3. Le mouvement reste synchronisé en continu avec la broche, même 'au delà' de la coordonnée du point final spécifié pendant l'arrêt de la broche et son inversion.
4. Le mouvement synchronisé se poursuit pour revenir aux coordonnées initiales.
5. Quand les coordonnées initiale sont atteintes, la commande inverse la broche une seconde fois (ex: de 300tr/mn sens anti-horaire à 300tr/mn en sens horaire)
6. Le mouvement reste synchronisé même 'au delà' des coordonnées initiales pendant que la broche s'arrête, puis s'inverse.
7. Un mouvement *non synchronisé* ramène le mobile en arrière, aux coordonnées initiales.

Tous les mouvements avec broche synchronisée ont besoin d'un index de broche, pour conserver la trajectoire prévue et que les passes se chevauchent exactement. Un mouvement avec *G33.1* se termine aux coordonnées initiales. Les mots d'axes sont facultatifs, sauf au moins un qui doit être utilisé.

**Exemple avec G33.1:**

```
G90 (mode distance absolue)
G0 X1.000 Y1.000 Z0.100 (mouvement rapide au point de édpart taraudage rigide
en 20 filets par pouce)
G33.1 Z-0.750 K0.05 (et une profondeur de filet de 0.750)
M2 (fin de programme)
```

– Voir les sections G90, G0 et M2 pour plus d'informations.

C'est une erreur si:
– Tous les axes sont omis.
– La broche ne tourne pas quand cette commande est exécutée.
– Le mouvement linéaire requis excède les limites de vitesse machine en raison d'une vitesse de broche trop élevée.

## 15.23 G38.x Mesure au palpeur

```
G38.x axes
```

– *G38.2* - palpe vers la pièce, stoppe au toucher, signale une erreur en cas de défaut.
– *G38.3* - palpe vers la pièce, stoppe au toucher.
– *G38.4* - palpe en quittant la pièce, stoppe en perdant le contact, signal une erreur en cas de défaut.
– *G38.5* - palpe en quittant la pièce, stoppe en perdant le contact.

**Important**
Cette commande n'est pas utilisable si la machine n'a pas été configurée pour exploiter un signal de sonde entre HAL et LinuxCNC. Le signal de la sonde doit être envoyé sur une broche d'entrée puis transmis à *motion.probe-entrée (bit, In)*. G38.x utilise la valeur de cette broche pour déterminer quand la sonde a touché ou perdu le contact. TRUE si le contact de la sonde est fermé (Touché), FALSE si il est ouvert.

Programmer *G38.x axes*, pour effectuer une mesure au palpeur. Les mots d'axe sont facultatifs excepté au moins un. Les mots d'axe définissent ensemble, le point de destination, à partir de l'emplacement actuel, vers lequel la sonde se déplace. Si le palpeur n'a pas déclenché avant que la destination soit atteinte, G38.2 et G38.4 signaleront une erreur. L'outil dans la broche doit être un palpeur ou un actionneur de contact.

En réponse à cette commande, la machine déplace le point contrôlé (qui est le centre de la boule du stylet du palpeur) en ligne droite, à la vitesse travail courante, vers le point programmé. En mode vitesse inverse du temps, la vitesse est telle que le mouvement depuis le point courant jusqu'au point programmé, prendra le temps spécifié. Le mouvement s'arrête (dans les limites d'accélération de la machine) lorsque le point programmé est atteint ou quand l'entrée du palpeur bascule dans l'état attendu selon la première éventualité.

Le tableau de signification des différents codes de mesure.

TABLE 15.2: Codes de mesure

| Code | État ciblé | Sens de destination | Signal d'erreur |
|---|---|---|---|
| G38.2 | Touché | Vers la pièce | Oui |
| G38.3 | Touché | Vers la pièce | Non |
| G38.4 | Quitté | Depuis la pièce | Oui |
| G38.5 | Quitté | Depuis la pièce | Non |

Après une mesure réussie, les paramètres 5061 à 5069 contiendront les coordonnées des axes XYZABCUVW, pour l'emplacement du point contrôlé à l'instant du changement d'état du palpeur. Après une mesure manquée, ils contiendront les coordonnées du point programmé. Le paramètre 5070 est mis à 1 si la mesure est réussie et à 0 si elle est manquée. Si la mesure n'a pas réussi, G38.2 et G38.4 signaleront une erreur en affichant un message à l'écran si l'interface graphique choisie le permet.

Un commentaire de la forme *(PROBEOPEN filename.txt)* ouvrira le fichier *filename.txt* et y enregistrera les 9 coordonnées de XYZABCUVW pour chaque mesure réussie. Le fichier doit être fermé avec le commentaire *(PROBECLOSE)*.

Dans le répertoire des exemples, le fichier *smartprobe.ngc* montre l'utilisation d'un palpeur et l'enregistrement des coordonnées de la pièce dans un fichier. Le fichier *smartprobe.ngc* peut être utilisé par *ngcgui* avec un minimum de modifications.

C'est une erreur si:
– Le point programmé est le même que le point courant.
– Aucun mot d'axe n'est utilisé.
– La compensation de d'outil est activée.
– La vitesse travail est à zéro.
– Le palpeur est déjà au contact de la cible.

## 15.24 G40 Révocation de la compensation de rayon d'outil

– *G40* - révoque la compensation de rayon d'outil. Le mouvement suivant, de sortie de compensation, doit être une droite au moins aussi longue que le diamètre de l'outil. Ce n'est pas une erreur de désactiver la compensation quand elle est déjà inactive.

**Exemple avec G40**

```
; la position courante est X1 èaprs la fin du mvt avec compensation
G40 (érvoque la compensation)
G0 X1.6 (mouvement élinaire aussi long que le èdiamtre d'outil)
M2 (fin de programme)
```

– Voir les sections G0 et M2 pour plus d'informations.

C'est une erreur si:
– Un mouvement en arc avec G2 ou G3 suit un G40.
– Le mouvement suivant la révocation de compensation est inférieur au diamètre de l'outil.

## 15.25 G41, G42 Compensation de rayon d'outil

```
G41 <D-> (compensation à gauche du profil)
G42 <D-> (compensation à droite du profil)
```

– *D* - Numéro d'outil

Le mot D est facultatif. En son absence ou si il est à zéro, le rayon de l'outil courant est utilisé. Si le mot D est présent, il devrait normalement correspondre au numéro de l'outil monté dans la broche, bien que cela ne soit pas indispensable, il doit par contre correspondre à un numéro d'outil valide.

Pour activer la compensation d'outil à gauche du profil, programmer *G41*. G41 applique la compensation d'outil à gauche de la ligne programmée vu de l'extrémité positive de l'axe perpendiculaire au plan.

Pour activer la compensation d'outil à droite du profil, programmer *G42*. G42 applique la correction d'outil à droite de la ligne programmée vu de l'extrémité positive de l'axe perpendiculaire au plan.

Le mouvement d'entrée doit être au moins aussi long que le rayon de l'outil. Le mouvement d'entrée peut être effectué en vitesse rapide.

La compensation d'outil ne peut être effectuée que si le plan XY ou le plan XZ est actif.

Les commandes définies par l'utilisateur, M100 à M199, sont autorisées lorsque la compensation d'outil est activée.

Le comportement de la machine, quand la compensation d'outil est activée, est décrit dans la section sur la compensation d'outil.

C'est une erreur si:
– Le nombre D ne correspond, ni à zéro, ni à un numéro d'outil valide.
– Le plan YZ est le plan de travail actif.
– La compensation d'outil est activée alors qu'elle est déjà active.

## 15.26 G41.1, G42.1 Compensation dynamique d'outil

```
G41.1 D- <L-> à( gauche du profil)
G42.1 D- <L-> à( droite du profil)
```

– Le mot D spécifie le diamètre de l'outil.
– Le mot L spécifie l'orientation de l'outil, est à 0 par défaut si non spécifié.

Pour activer la compensation dynamique d'outil à gauche du profil, programmer *G41.1 D- L-*.

Pour activer la compensation dynamique d'outil à droite du profil, programmer *G42.1 D- L-*.

C'est une erreur si:

– Le plan YZ est le plan de travail actif.
– La valeur de L n'est pas comprise entre 0 et 9 inclus.
– Le nombre L est utilisée alors que le plan XZ n'est pas le plan actif.
– La compensation d'outil est activée alors qu'elle est déjà active.

Plus d'informations sur l'orientation des outils, sur les outils de tour en 1-2-3-4 et les outils de tour en 5-6-7-8.

## 15.27 G43 Activation de la compensation de longueur d'outil

– *H* - Numéro d'outil
– *G43* - Utilise l'outil courant chargé par le dernier Tn M6. G43 modifie les mouvements ultérieurs en décalant les coordonnées de Z et/ou de X, de la longueur de l'outil. G43 ne provoque aucun mouvement. L'effet de la compensation ne se produira qu'au cours du prochain mouvement des axes compensés, de sorte que le point final de ce mouvement sera la position compensée.
– *G43 H-* - Utilise l'offset de l'outil correspondant fourni par la table d'outils. Ce n'est pas une erreur d'avoir la valeur de H à zéro, le numéro de l'outil courant sera utilisé.

**Exemple de ligne avec G43 H-**

```
G43 H1 (ajuste les offsets d'outil avec les valeurs de l'outil 1 fournies par
la table d'outils)
```

C'est une erreur si:
– La valeur de H n'est pas un entier, il est négatif, ou il ne correspond, ni à zéro, ni à un numéro d'outil valide.

## 15.28 G43.1 Compensation dynamique de longueur d'outil

```
G43.1 axes
```

– *G43.1 axes* - Modifie les mouvements ultérieurs en décalant les coordonnées de Z et/ou de X, selon les offsets stockés dans la table d'outils. G43.1 ne provoque aucun mouvement. L'effet de la compensation ne se produira qu'au cours du prochain mouvement des axes compensés de sorte que le point final de ce mouvement sera la position compensée.

**Exemple avec G43.1**

```
G90 (passe en mode absolu)
T1 M6 G43 (charge l'outil °N1 et son offset de longueur, Z est à la position
machine 0 et la visu affiche Z1.500)
G43.1 Z0.250 (édcale l'outil courant de 0.250, la visu affiche maintenant
Z1.250)
M2 (fin de programme)
```

– Voir les sections G90 & T et M2 pour plus d'informations.

C'est une erreur si:
– Une commande de mouvement est sur la même ligne que *G43.1*

## 15.29 G49 Révocation de la compensation de longueur d'outil

Pour révoquer la compensation de longueur d'outil, programmer *G49*.

Ce n'est pas une erreur de programmer une compensation qui est déjà utilisée. Ce n'est pas non plus une erreur de révoquer une compensation de longueur d'outil alors qu'aucune n'est couramment utilisée.

## 15.30 G53 Mouvement en coordonnées absolues

```
G53 axes
```

Pour un déplacement exprimé en coordonnées système, programmer *G53* sur la même ligne qu'un mouvement linéaire. *G53* n'est pas modal, il doit donc être programmé sur chaque ligne où il doit être actif. *G0* ou *G1* ne doivent pas se trouver sur la même ligne si un d'eux est déjà actif. Par exemple:

**Exemple avec G53**

```
G53 G0 X0 Y0 Z0 (mouvement élinaire rapide des axes à leur positions d'origine)
G53 X2 (mouvement élinaire rapide à la écoordonne absolue X=2)
```

C'est une erreur si:
– *G53* est utilisé sans que G0 ou G1 ne soit actif.
– *G53* est utilisé alors que la compensation d'outil est active.
Étudier le chapitre sur les systèmes de coordonnées et de leurs décalages, pour bien maîtriser ces concepts.

## 15.31 G54 à G59.3 Choix du système de coordonnées

– *G54* - Système de coordonnées pièce 1
– *G55* - Système de coordonnées pièce 2
– *G56* - Système de coordonnées pièce 3
– *G57* - Système de coordonnées pièce 4
– *G58* - Système de coordonnées pièce 5
– *G59* - Système de coordonnées pièce 6
– *G59.1* - Système de coordonnées pièce 7
– *G59.2* - Système de coordonnées pièce 8
– *G59.3* - Système de coordonnées pièce 9

Le code *G54* est apparié avec le système de coordonnées pièce N°1, pour le choisir programmer *G54* et ainsi de suite pour les autres systèmes.

Les systèmes de coordonnées stockent les valeurs de chacun des axes dans les variables indiquées dans le tableau ci-dessous.

TABLE 15.3: Paramètres des systèmes de coordonnées pièce

| Choix | CS | X | Y | Z | A | B | C | U | V | W | R |
|---|---|---|---|---|---|---|---|---|---|---|---|
| G54 | 1 | 5221 | 5222 | 5223 | 5224 | 5225 | 5226 | 5227 | 5228 | 5229 | 5230 |
| G55 | 2 | 5241 | 5242 | 5243 | 5244 | 5245 | 5246 | 5247 | 5248 | 5249 | 5250 |
| G56 | 3 | 5261 | 5262 | 5263 | 5264 | 5265 | 5266 | 5267 | 5268 | 5269 | 5270 |
| G57 | 4 | 5281 | 5282 | 5283 | 5284 | 5285 | 5286 | 5287 | 5288 | 5289 | 5290 |
| G58 | 5 | 5301 | 5302 | 5303 | 5304 | 5305 | 5306 | 5307 | 5308 | 5309 | 5310 |
| G59 | 6 | 5321 | 5322 | 5323 | 5324 | 5325 | 5326 | 5327 | 5328 | 5329 | 5330 |
| G59.1 | 7 | 5341 | 5342 | 5343 | 5344 | 5345 | 5346 | 5347 | 5348 | 5349 | 5350 |
| G59.2 | 8 | 5361 | 5362 | 5363 | 5364 | 5365 | 5366 | 5367 | 5368 | 5369 | 5370 |
| G59.3 | 9 | 5381 | 5382 | 5383 | 5384 | 5385 | 5386 | 5387 | 5388 | 5389 | 5390 |

C'est une erreur si:
– Un de ces G-codes est utilisé alors que la compensation d'outil est active.
Voir la section sur les systèmes de coordonnée pour une vue complète.

## 15.32 G61, G61.1 Contrôle de trajectoire exacte

– *G61* - Met la machine en mode de trajectoire exacte. G61 suivra exactement la trajectoire programmée même si cela doit aboutir à un arrêt complet momentané du mobile.
– *G61.1* - Met la machine en mode arrêts exacts.

## 15.33 G64 Contrôle de trajectoire continue avec tolérance

```
G64 <P- <Q->>
```

– *P-* - Déviation maximale tolérée par rapport à la trajectoire programmée.
– *Q-* - Tolérance naïve cam.
– *G64* - Recherche de la meilleure vitesse possible.
– *G64 P-* - Mélange entre meilleure vitesse et tolérance de déviation.
– *G64 P- Q-* - Est le moyen d'affiner encore pour obtenir le meilleur compromis entre vitesse et précision de la trajectoire. La vitesse sera réduite si nécessaire pour maintenir la trajectoire, même si ça doit aboutir à un arrêt complet momentané. Le *détecteur naïve cam* est activé. Quand il y a une série de mouvements linéaires XYZ en vitesse travail, avec une même vitesse de déplacement, inférieure à *Q-*, ils sont regroupés en un seul segment linéaire, ainsi la vitesse s'en trouve améliorée puisqu'il n'y a plus de décélération/arrêt/accélération aux points de jonction des segments. Sur les mouvements G2/G3 dans le plan *G17* (XY) lorsque le maximum d'écart entre un arc et une ligne droite est inférieur à la déviation maximale *P-*, la tolérance de l'arc est divisée en deux lignes (depuis le début de l'arc jusqu'au milieu et du milieu jusqu'à la fin). Ces deux lignes sont ensuite soumises à l'algorithme *naïve cam*. Ainsi, les cas ligne-arc, arc-arc et arc-ligne et le cas ligne-ligne, bénéficient de l'algorithme *naïve cam*, ce qui améliore les performances en simplifiant les trajectoires. Il est permis de programmer ce mode même si il est déjà actif.

**Exemple de ligne de programme avec G64**

```
G64 P0.015 (fixe la édviation d'usinage à 0.015 maximum de la trajectoireé
programme)
```

Il est de bonne pratique de spécifier un type de contrôle de trajectoire dans le préambule de chaque programme G-code.

## 15.34 G73 Cycle de perçage avec brise copeaux

```
G73 axes R- Q- <L->
```

– *R-* - Position du plan de retrait en Z
– *Q-* - Incrément *delta* parallèle à l'axe Z
– *L-* - Répétition

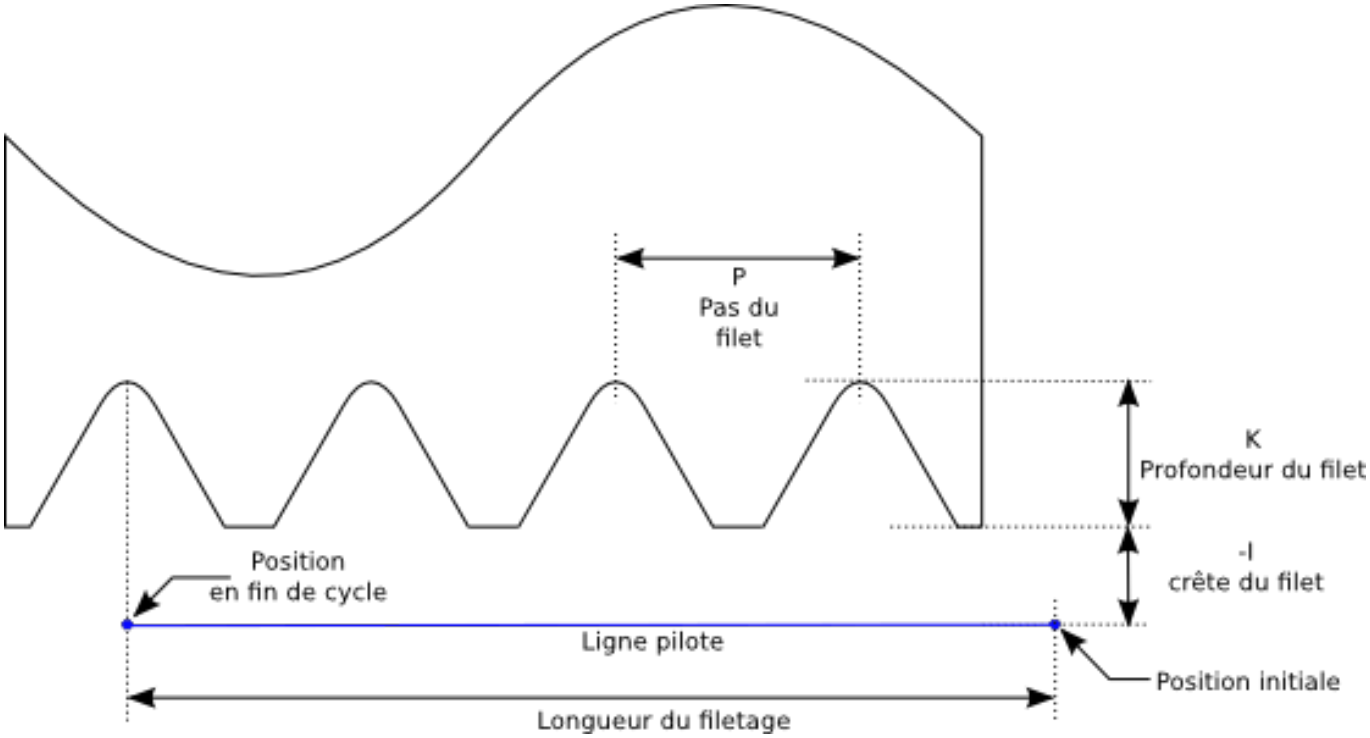
Le cycle *G73* est destiné au perçage profond ou au fraisage avec brise-copeaux. Les retraits, au cours de ce cycle, fragmentent les copeaux longs (fréquents lors de l'usinage de l'aluminium). Ce cycle utilise la valeur *Q-* qui représente un incrément *delta* parallèle à l'axe Z. Le cycle se décompose de la manière suivante:

1. Un mouvement préliminaire. Comme décrit dans cet exposé sur le mouvement préliminaire
2. Un mouvement de l'axe Z seul, en vitesse travail, sur la position la moins profonde entre, l'incrément *delta* ou la position de Z programmée.
3. Une petite remontée en vitesse rapide.
4. Répétition des étapes 2 et 3 jusqu'à ce que la position programmée de Z soit atteinte à l'étape 2.
5. Un mouvement de l'axe Z en vitesse rapide jusqu'au plan de retrait.

C'est une erreur si:
– La valeur de Q est négative ou égale à zéro.
– Le nombre R n'est pas spécifié.

## 15.35 G76 Cycle de filetage préprogrammé

```
G76 P- Z- I- J- R- K- Q- H- E- L-
```




– *Ligne pilote* - La ligne pilote est une ligne imaginaire, parallèle à l'axe de la broche (Z), située en sécurité à l'extérieur du matériau à fileter. La ligne pilote va du point initial en Z jusqu'à la fin du filetage donnée par la valeur de *Z* dans la commande.
– *P-* - Le pas du filet en distance de déplacement par tour.
– *Z-* - La position finale du filetage. A la fin du cycle, l'outil sera à cette position *Z*.

---

**Note**
En mode diamètre G7, les valeurs *I*, *J* et *K* sont des mesures de diamètre. En mode rayon G8, les valeurs *I*, *J* et *K* sont des mesures de rayon.

---

– *I-* - La crête du filet est une distance entre la ligne pilote et la surface de la pièce. Une valeur négative de *I*, indique un filetage externe et une valeur positive, indique un filetage interne. C'est généralement à ce diamètre nominal que le matériau est cylindré avant de commencer le cycle *G76*.
– *J-* - Une valeur positive, spécifie la profondeur de la passe initiale. La première passe sera à *J* au delà de la crête du filet *I*.
– *K-* - Une valeur positive, spécifie la profondeur finale du filet. La dernière passe du filetage sera à *K* au delà de la crête du filet *I*.

Paramètres facultatifs:
– *R-* - La profondeur de dégressivité. *R1.0* spécifie une profondeur de passe constante pour les passes successives du filetage. *R2.0* spécifie une surface constante. Les valeurs comprises entre 1.0 et 2.0 spécifient une profondeur décroissante mais une surface croissante. Enfin, les valeurs supérieures à 2.0 sélectionnent une surface décroissante.

---

**AVERTISSEMENT**
Les valeurs inutilement hautes de dégressivité, produiront un nombre inutilement important de passes. (dégressivité = plongée par paliers)

---

– *Q*- - L'angle de pénétration oblique. C'est l'angle (en degrés) décrivant de combien, les passes successives doivent être décalées le long de l'axe Z. C'est utilisé pour faire enlever plus de matériau d'un côté de l'outil que de l'autre. Une valeur positive de *Q* fait couper d'avantage le bord de l'outil. Typiquement, les valeurs sont 29, 29.5 ou 30 degrés.
– *H*- - Le nombre de passes de finition. Les passes de finition sont des passes additionnelles en fond de filet. Pour ne pas faire de passe de finition, programmer *H0*.

Les entrées et sorties de filetage peuvent être programmées coniques avec les valeurs de *E* et *L*.
– *E*- - Spécifie la longueur des parties coniques le long de l'axe Z. L'angle du cône ira de la profondeur de la dernière passe à la crête du filet *I*. *E2.0* donnera un cône d'entrée et de sortie d'une longueur de 2.0 unités dans le sens du filetage. Pour un cône à 45 degrés, programmer *E* identique à *K*.
– *L*- - Spécifie quelles extrémités du filetage doivent être coniques. Programmer *L0* pour aucune (par défaut), *L1* pour une entrée conique, *L2* pour une sortie conique, ou *L3* pour l'entrée et la sortie coniques.
L'outil fera une brève pause pour la synchronisation avec l'impulsion d'index avant chaque passe de filetage. Une gorge de dégagement sera requise à l'entrée, à moins que le début du filetage ne soit après l'extrémité de la pièce ou qu'un cône d'entrée soit utilisé.

À moins d'utiliser un cône de sortie, le mouvement de sortie (retour rapide sur X initial) n'est pas synchronisé sur la vitesse de broche. Avec une broche lente, la sortie pourrait se faire sur une petite fraction de tour. Si la vitesse de broche est augmentée après qu'un certain nombre de passes soient déjà faites, la sortie va prendre une plus grande fraction de tour, il en résultera un usinage *très brutal* pendant ce nouveau mouvement de sortie. Ceci peut être évité en prévoyant une gorge de sortie, ou en ne changeant pas la vitesse de broche pendant le filetage.

La position finale de l'outil sera à la fin de la *ligne pilote*. Un mouvement de sécurité peut être nécessaire avec un filetage interne, pour sortir l'outil de la pièce.

C'est une erreur si:
– Le plan de travail actif n'est pas ZX.
– D'autres mots d'axes que X ou Y, sont spécifiés.
– La dégressivité *R* est inférieure à 1.0.
– Tous les mots requis ne sont pas spécifiés.
– *P*, *J*, *K* ou *H* est négatif.
– *E*- est supérieur à la moitié de la longueur de la ligne pilote.
**Connections de HAL** Les pins *motion.spindle-at-speed* et l'index *encoder.n.phase-Z* doivent être connectées dans le fichier HAL pour que G76 soit opérationnel. Voir le Manuel de l'intégrateur pour plus d'informations sur les mouvements synchronisés avec la broche.

**Informations techniques** Le cycle préprogrammé G76 est basé sur le mouvement avec broche synchronisée G33, voir les informations technique relatives à G33.

Un programme de filetage, *g76.ngc* montre l'utilisation d'un cycle de filetage G76, il peut être visualisé et exécuté sur n'importe quelle machine utilisant la configuration *sim/lathe.ini*.

**Exemple de G-Code avec G76**

```
G0 Z-0.5 X0.2
G76 P0.05 Z-1 I-0.075 J0.008 K0.045 Q29.5 L2 E0.045
```

Sur l'image ci-dessous, l'outil est à la position finale après que le cycle G76 soit terminé. On voit que le parcours d'entrée de l'outil sur la droite, spécifié par Q29.5 et le parcours de sortie conique à gauche comme spécifié par L2 E0.045. Les lignes blanches sont les mouvements de coupe.

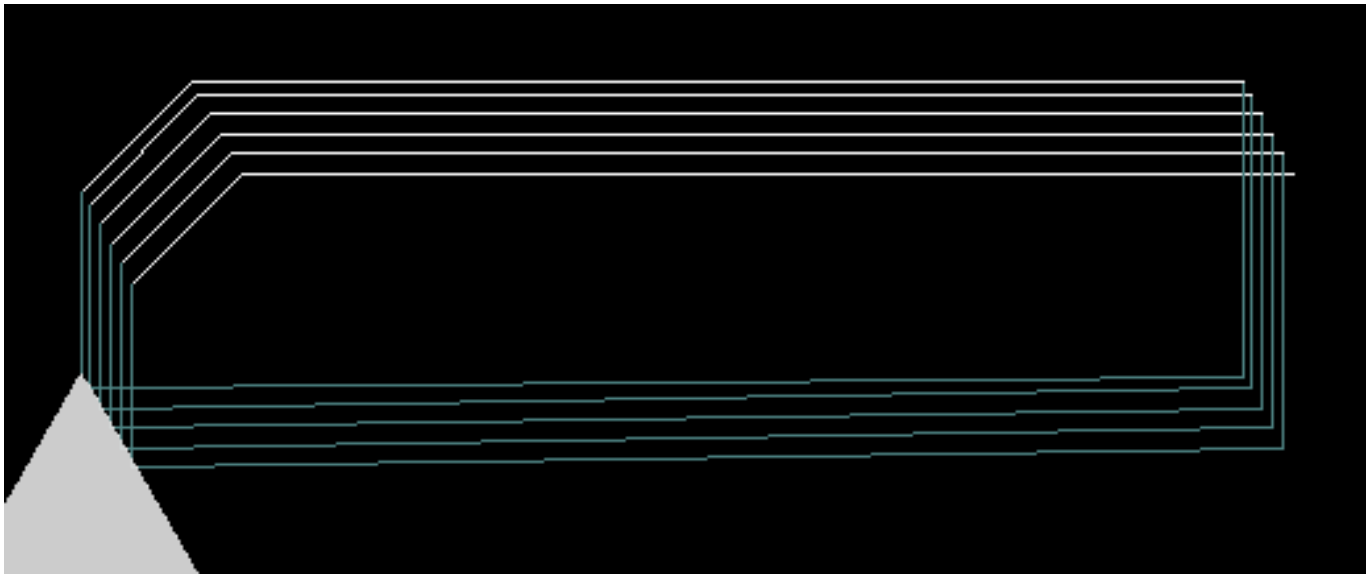


FIGURE 15.3 – Parcours d'outil de l'exemple

## 15.36 Les cycles de perçage G81 à G89

Les cycles de perçage de *G81* à *G89* et la révocation de ces cycle *G80*, sont décrits dans cette section. Des exemples sont donnés plus bas avec les descriptions.

Tous les cycles de perçage sont effectués dans le respect du plan de travail courant. N'importe lequel des six plans de travail peut être choisi. Dans cette section, la plupart des descriptions supposeront que le plan de travail XY est le plan courant. Le comportement reste analogue pour les autres plans de travail et les mots corrects doivent être utilisés. Par exemple, dans le plan G17.1, l'action de retrait s'effectue parallèlement à l'axe W et les positions ou incréments sont donnés avec U et W. Dans ce cas, substituer U, V, W avec X, Y, Z dans les instructions suivantes.

Les mots d'axes rotatifs ne sont pas autorisés dans les cycles de perçage. Quand le plan actif est X, Y, Z, les mots d'axes U, V, W ne sont pas autorisés. De même, si le plan actif est U, V, W, les mots d'axes X, Y, Z ne sont pas autorisés.

### 15.36.1 Mots communs

Tous les cycles de perçage utilisent les groupes X, Y, Z ou U, V, W selon le plan sélectionné, ainsi que le mot *R*. La position de R- (signifiant retrait) est perpendiculaire au plan de travail courant (axe Z pour le plan XY, axe X pour le plan YZ, axe Y pour le plan XZ, etc.). Quelques cycles de perçage utilisent des arguments supplémentaires.

### 15.36.2 Mots *sticky*

Dans les cycles de perçage, un nombre est qualifié de *sticky* (persistante, collant) si, quand le même cycle est répété sur plusieurs lignes de code en colonne, le nombre doit être indiqué la première fois, mais il devient facultatif pour le reste des lignes suivantes. Les nombres *sticky* conservent leur valeur tant qu'ils ne sont pas explicitement programmés avec une nouvelle valeur. La valeur de R est toujours *sticky*.

En mode de déplacements incrémentaux (G91), les valeurs X, Y, est R sont traitées comme des incréments depuis la position courante, Z est un incrément depuis la position de l'axe Z avant le mouvement impliquant l'axe Z. En mode de déplacements absolus, les valeurs de X, Y, R, et Z sont des positions absolues dans le système de coordonnées courant.

### 15.36.3 Répétition de cycle

Le mot L est facultatif et représente le nombre de répétitions. L=0 n'est pas permis. Si les fonctionnalités de répétition sont utilisées, elles le sont normalement en mode relatif, de sorte que la même séquence de mouvements se répète à plusieurs emplacements régulièrement espacés le long d'une ligne droite. Quand L>1 en mode relatif et XY comme plan courant, les positions X et Y sont déterminées en ajoutant les valeurs X et Y de la commande à celles de la position courante, pour le premier trajet ou ensuite, à celles de la position finale du précédent trajet, pour les répétitions. Ainsi, si vous programmez `L10`, vous obtiendrez 10 cycles. Le premier cycle sera la distance X, Y depuis la position d'origine. Les positions de R- et Z- ne changent pas durant toutes les répétitions. En mode absolu, L>1 signifie `faire le même cycle à la même place plusieurs fois`, omis, le mot L est équivalent à L=1. La valeur de L n'est pas *sticky*.

### 15.36.4 Mode de retrait

La hauteur du mouvement de retrait à la fin de chaque répétition (appelée *plan de retrait* dans les descriptions suivantes) est déterminée par le mode de retrait: retrait sur la position initiale de Z, si elle est au dessus de la valeur de R et que le mode de retrait est *G98*, OLD_Z, sinon, à la position de R. Voir la section sur les options du plan de retrait.

### 15.36.5 Erreurs des cycles de perçage

Il y a une erreur si:
- Tous les mots X, Y et Z sont manquants durant un cycle de perçage.
- Des mots d'axes de différents groupes (XYZ) (UVW) sont utilisés.
- Un nombre P est requis mais un nombre P négatif est utilisé.
- Un nombre L est utilisé mais n'est pas un entier positif.
- Un mouvement d'axe rotatif est utilisé durant un cycle de perçage.
- Une vitesse inverse du temps est activée durant un cycle de perçage.
- La compensation d'outil est activée durant un cycle de perçage.

Quand le plan XY est actif, la valeur de Z est *sticky*, et c'est une erreur si:
- La valeur de Z est manquante alors qu'un même cycle de perçage n'a pas encore été activé.
- La valeur de R est inférieure à celle de Z.

Si un autre plan est actif, les conditions d'erreur sont analogues à celles du plan XY décrites ci-dessus.

### 15.36.6 Mouvement préliminaire et Intermédiaire

Le mouvement préliminaire est un ensemble de mouvements commun à tous les cycles de perçage.

Tout au début de l'exécution d'un cycle de perçage, si la position actuelle de Z est en dessous de la position de retrait R, l'axe Z va à la position R. Ceci n'arrive qu'une fois, sans tenir compte de la valeur de L.

En plus, au début du premier cycle et à chaque répétition, un ou deux des mouvements suivants sont faits:

1. Un déplacement en ligne droite, parallèle au plan XY, vers le position programmée.
2. Un déplacement en ligne droite, de l'axe Z seul vers la position de retrait R, si il n'est pas déjà à cette position R.

Si un autre plan est actif, le mouvement préliminaire et intermédiaire est analogue.

### 15.36.7 Pourquoi utiliser les cycles de perçage?

Il y a au moins deux raisons pour utiliser les cycles de perçage. La première est l'économie de code et la seconde la sécurité offerte par le mouvement préliminaire qui permet de ne pas s'occuper du point de départ du cycle.

## 15.37 G80 Révocation des codes modaux

– *G80* - Révoque, tant qu'il est actif, tous les codes de mouvements modaux du groupe 1 auquel il appartient. Il est révoqué lui même par tout g-code du même groupe.

C'est une erreur si:
– Des mots d'axes sont programmés quand G80 est actif.
**Exemple 1 avec G80:**

```
G90 G81 X1 Y1 Z1.5 R2.8 (cycle de çperage en mode de édplacement absolu)
G80 (érvoque G81)
G0 X0 Y0 Z0 (active les mouvements en vitesse rapide et édplace le
mobile en X0, Y0 et Z0)
```

L'exemple 1 produit les mêmes déplacements et le même état final de la machine que l'exemple suivant:

**Exemple avec G0:**

```
G90 G81 X1 Y1 Z1.5 R2.8 (cycle de çperage en mode de édplacement absolu)
G0 X0 Y0 Z0 (active les mouvements en vitesse rapide et édplace le
mobile en X0, Y0 et Z0)
```

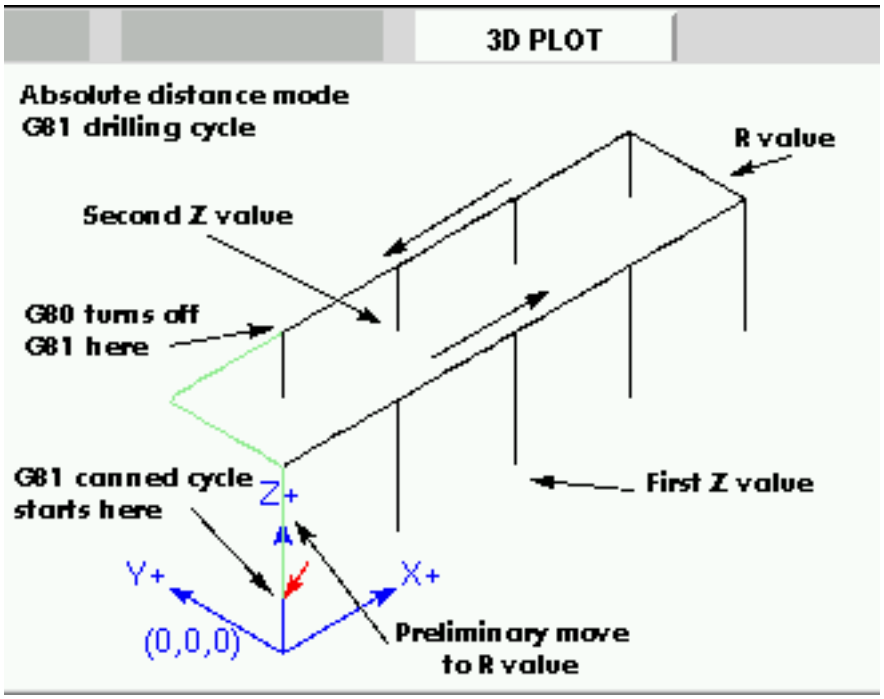
L'avantage du premier exemple est que la ligne du G80 révoque clairement le cycle G81. Avec ce premier programme, le programmeur doit revenir en mode mouvement avec G0, comme c'est fait sur la ligne suivante, ou tout autre mot G de mouvement.

Si un cycle de perçage n'est pas révoqué avec G80 ou un autre mot G de mouvement, le cycle de perçage attend de se répéter en utilisant la prochaine ligne de code contenant un ou plusieurs mots d'axe. Le fichier suivant perce (G81) un ensemble de huit trous, tel que montré sur l'image qui suit.

**Exemple 2 avec G80:**

```
N100 G90 G0 X0 Y0 Z0 (écoordonnes d'origine)
N110 G1 X0 G4 P0.1
N120 G81 X1 Y0 Z0 R1 (cycle de çperage)
N130 X2
N140 X3
N150 X4
N160 Y1 Z0.5
N170 X3
N180 X2
N190 X1
N200 G80 (érvocation du cycle G81)
N210 G0 X0 (mouvement en vitesse rapide)
N220 Y0
N230 Z0
N240 M2 (fin du programme)
```

**Note**
Noter que la position de Z change après les quatre premiers trous. C'est également un des rares cas dans lesquels les numéros de lignes sont présents, permettant d'envoyer le lecteur sur une ligne de code spécifique.




L'utilisation du G80 de la ligne N200 est facultative puisqu'il y a un G0 sur la ligne suivante qui révoque le cycle G81. Mais utiliser G80, comme l'exemple 2 le montre, donne une meilleure lisibilité au programme. Sans ce G80, il ne serait pas aussi évident que tous les blocs compris entre N120 et N200 appartiennent au cycle de perçage.

## 15.38 G81 Cycle de perçage

```
G81 (X- Y- Z- ) ou (U- V- W- ) R- L-
```

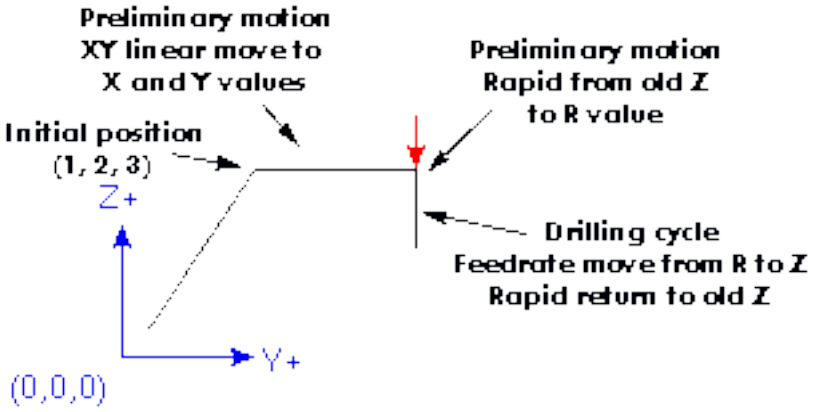
Le cycle *G81* est destiné au perçage.

1. Un mouvement préliminaire, comme décrit sur cette page.
2. Un déplacement de l'axe Z seul à la vitesse programmée, vers la position Z programmée.
3. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

**Exemple 1: G81 en position absolue** Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de code suivante est interprétée:

```
G90 G81 G98 X4 Y5 Z1.5 R2.8
```

Le mode de déplacements absolus est appelé *(G90)*, le plan de retrait est positionné sur OLD_Z *(G98)*, l'appel du cycle de perçage *G81* va lancer ce cycle une fois. La position X deviendra celle demandée, X4. La position de Y deviendra celle demandée, Y5. La position de Z deviendra celle demandée, Z1.5. La valeur de R fixe le plan de retrait de Z à 2.8. La valeur de OLD_Z est 3. Les mouvements suivants vont se produire.




– Un mouvement en vitesse rapide, parallèle au plan XY vers X4, Y5, Z3
– Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z2.8
– Un mouvement en vitesse travail, parallèle à l'axe Z vers X4, Y5, Z1.5
– Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z3

*Exemple 2:* Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de codes suivante est interprétée:

```
G91 G81 G98 X4 Y5 Z-0.6 R1.8 L3
```

Le mode de déplacements incrémentaux est appelé *(G91)*, le plan de retrait est positionné sur OLD_Z *(G98)*, l'appel du cycle de perçage *G81* demande 3 répétitions du cycle. La valeur demandée de X est 4, la valeur demandée de Y est 5, la valeur demandée de Z est -0.6 et le retrait R est à 1.8. La position initiale de X sera 5 (1+4), la position initiale de Y sera 7 (2+5), le plan de retrait sera positionné sur 4.8 (1.8+3) et Z positionné sur 4.2 (4.8-0.6). OLD_Z est à 3.

Le premier mouvement en vitesse rapide le long de l'axe Z vers X1, Y2, Z4.8), puisque OLD_Z est inférieur au plan de retrait.

La première répétition produira 3 mouvements.

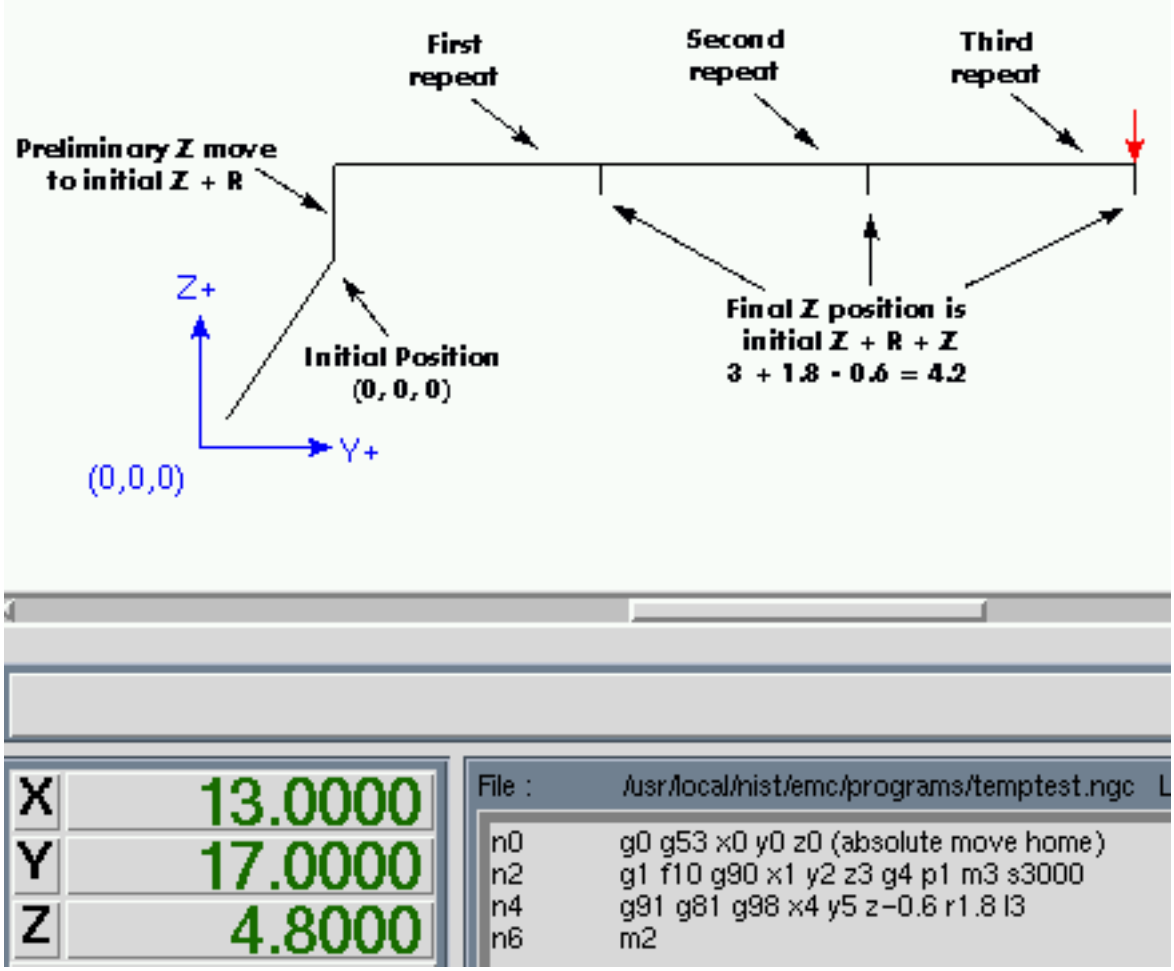
1. Un déplacement en vitesse rapide, parallèle au plan XY vers X5, Y7, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X5, Y7, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X5, Y7, Z4.8

La deuxième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 9, la position de Y est augmentée de 5 et passe à 12.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X9, Y12, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X9, Y12, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X9, Y12, Z4.8

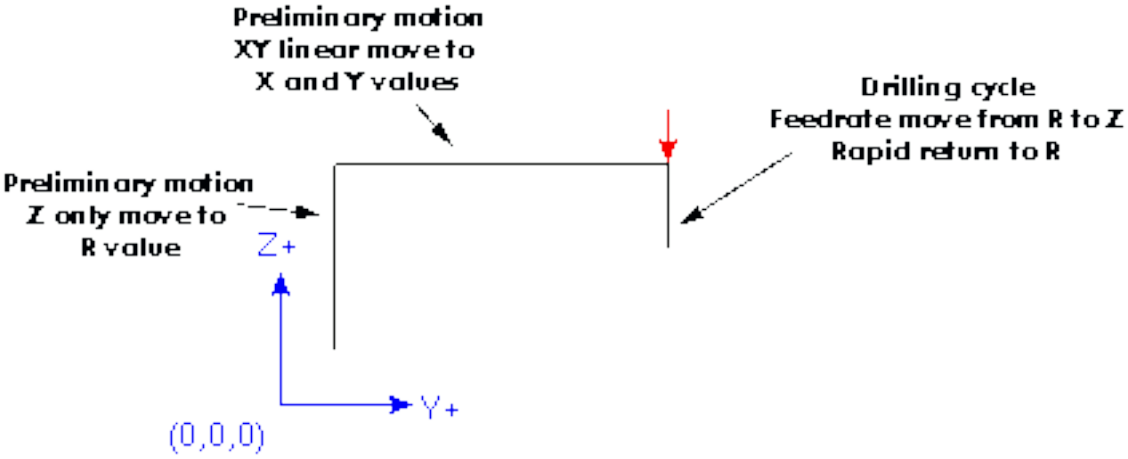
La troisième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 13, la position de Y est augmentée de 5 et passe à 17.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X13, Y17, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X13, Y17, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X13, Y17, Z4.8




*Exemple 3:* G81 en position relative

Supposons maintenant que le premier g81 de la ligne de code soit exécuté, mais de (0, 0, 0) plutôt que de (1, 2, 3). G90 G81 G98 X4 Y5 Z1.5 R2.8 Depuis OLD_Z est inférieur à la valeur de R, il n'ajoute rien au mouvement, mais puisque la valeur initiale de Z est inférieure à la valeur spécifiée dans R, un premier mouvement de Z sera effectué durant le mouvement préliminaire.

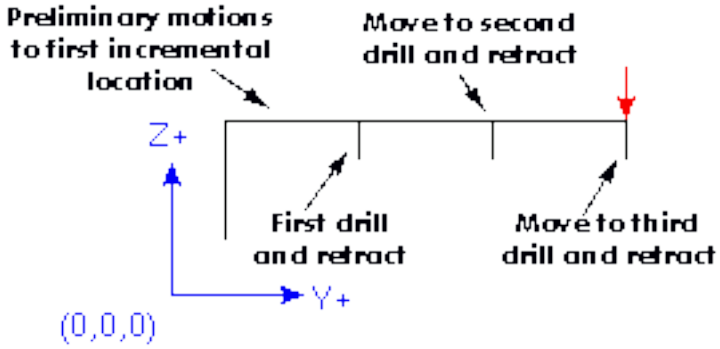



*Exemple 4:* G81 en absolu avec R > Z

Il s'agit de la trajectoire pour le second bloc de code de G81.

```
G91 G81 G98 X4 Y5 Z-0.6 R1.8 L3
```

Cette trajectoire commence en (0, 0, 0), l'interpréteur ajoute les valeurs initiales Z0 et R 1.8 et déplace le mobile en vitesse rapide vers cet emplacement. Après ce premier déplacement initial de Z, la répétition fonctionne de manière identique à celle de l'exemple 3 avec le mouvement final de Z à 0.6 en dessous de la valeur de R.




*Exemple 5:* G81 en relatif avec R > Z

```
G90 G98 G81 X4 Y5 Z-0.6 R1.8
```

Puisque ce tracé commence en (X0, Y0, Z0), l'interpréteur ajoute R1.8 au Z0 initial et déplace le mobile en vitesse rapide à cet emplacement, comme dans *l'exemple 4*. Après ce mouvement initial à une hauteur Z0.6, le mouvement en vitesse rapide se terminera en X4 Y5. Alors la hauteur Z sera à 0.6 en dessous de la valeur de R. La fonction de répétition fera encore déplacer Z au même emplacement.

## 15.39 G82 Cycle de perçage avec temporisation

```
G82 (X- Y- Z- ) ou (U- V- W- ) R- L- P-
```

Le cycle *G82* est destiné au perçage. Les mouvements du cycle G82 ressemblent à ceux de G81 avec une temporisation supplémentaire en fin de mouvement Z. La longueur de cette temporisation, exprimée en secondes, est spécifiée par un mot P# sur la ligne du G82.

1. Un mouvement préliminaire. Comme décrit sur cette page.
2. Un déplacement de l'axe Z seul en vitesse programmée, vers la position Z programmée.
3. Une temporisation de *P* secondes.
4. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait *R*.

```
G90 G82 G98 X4 Y5 Z1.5 R2.8 P2
```

Sera équivalent à l'exemple 3 ci-dessus mais avec une temporisation de 2 secondes en fond de trou.

## 15.40 G83 Cycle de perçage avec débourrage

```
G83 (X- Y- Z-) or (U- V- W-) R- L- Q-
```

Le cycle *G83* est destiné au perçage profond ou au fraisage avec brise-copeaux. Les retraits, au cours de ce cycle, dégagent les copeaux du trou et fragmentent les copeaux longs (qui sont fréquents lors du perçage dans l'aluminium). Ce cycle utilise la valeur *Q* qui représente un incrément *delta* le long de l'axe Z.

donnera:

1. Un mouvement préliminaire, comme décrit sur cette page.
2. Un mouvement de l'axe Z seul, en vitesse travail, sur la position la moins profonde entre, un incrément delta, ou la position de Z programmée.
3. Un mouvement en vitesse rapide au plan de retrait.
4. Une plongée en vitesse rapide dans le même trou, presque jusqu'au fond.
5. Répétition des étapes 2, 3 et 4 jusqu'à ce que la position programmée de Z soit atteinte à l'étape 2.
6. Un mouvement de l'axe Z en vitesse rapide vers le plan de retrait.

C'est une erreur si:
– La valeur de Q est négative ou égale à zéro.

## 15.41 G84 Cycle de taraudage à droite

Ce code n'est pas encore implémenté dans LinuxCNC. Il est accepté mais son comportement n'est pas défini. Voir le taraudage rigide.

## 15.42 G85 Cycle d'alésage, sans temporisation, retrait en vitesse travail

```
G85 (X- Y- Z-) or (U- V- W-) R- L-
```

Le cycle *G85* est destiné à l'alésage, mais peut être utilisé pour le perçage ou le fraisage.

1. Un mouvement préliminaire, comme décrit sur cette page.
2. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.
3. Retrait de l'axe Z en vitesse travail vers le plan de retrait.

## 15.43 G86 Cycle d'alésage, arrêt de broche, retrait en vitesse rapide

```
G86 (X- Y- Z-) or (U- V- W-) R- L- P-
```

Le cycle *G86* est destiné à l'alésage. Ce cycle utilise la valeur P pour une temporisation en secondes.

1. Un mouvement préliminaire, comme décrit sur cette page.
2. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.
3. Une temporisation de P secondes.
4. L'arrêt de rotation de la broche.
5. Retrait de l'axe Z en vitesse rapide vers le plan de retrait.
6. Reprise de la rotation de la broche dans la même direction que précédemment.

La broche doit tourner avant le lancement de ce cycle. C'est une erreur si:
– La broche ne tourne pas avant que ce cycle ne soit exécuté.

## 15.44 G87 Alésage inverse

Ce code n'est pas encore implémenté dans LinuxCNC. Il est accepté mais son comportement n'est pas défini.

## 15.45 G88 Alésage, arrêt de broche, retrait en manuel

Ce code n'est pas encore implémenté dans LinuxCNC. Il est accepté mais son comportement n'est pas défini.

## 15.46 G89 Cycle d'alésage, temporisation, retrait en vitesse travail

```
G89 (X- Y- Z-) or (U- V- W-) R- L- P-
```

Le cycle *G89* est destiné à l'alésage. Il utilise la valeur de P pour une temporisation en secondes.

1. Un mouvement préliminaire, comme décrit sur cette page.
2. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.
3. Temporisation de P secondes.
4. Retrait de l'axe Z en vitesse travail vers le plan de retrait.

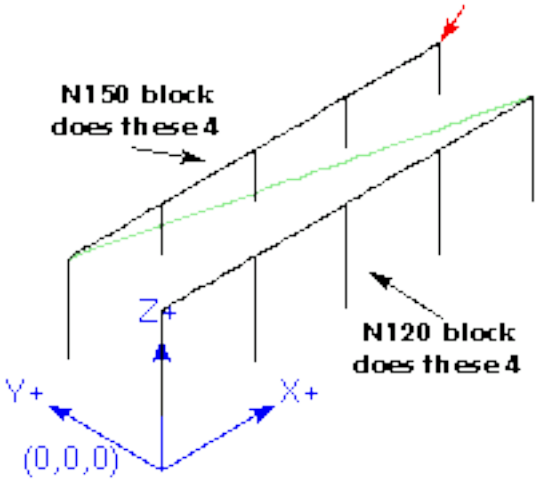
### 15.46.1 Pourquoi utiliser les cycles de perçage ?

Il y a au moins deux raisons, la première est l'économie de code. Un simple trou demande plusieurs lignes de code pour être exécuté.

Nous avons montré plus haut, comment les cycles de perçage peuvent être utilisés pour produire 8 trous avec dix lignes de code. Le programme ci-dessous permet de produire le même jeu de 8 trous en utilisant cinq lignes pour le cycle de perçage. Il ne suit pas exactement le même parcours et ne perce pas dans le même ordre que l'exemple précédent, mais le programme a été écrit de manière économique, une bonne pratique qui devrait être courante avec les cycles de perçage.

*Exemple 5:* perçage de huit trous, réécrit.

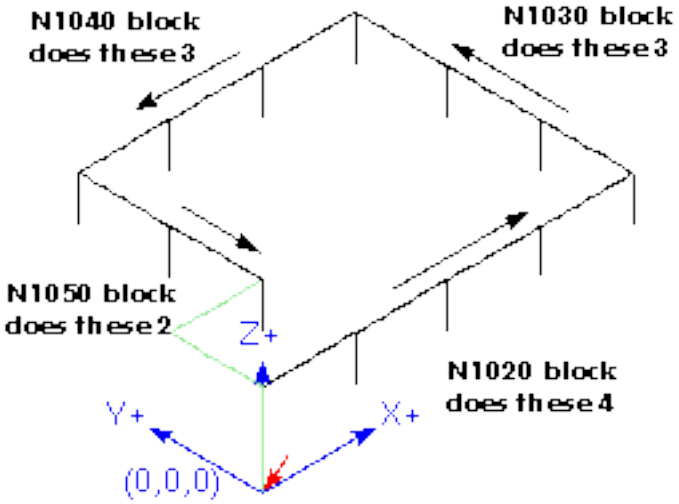
```
G90 G0 X0 Y0 Z0 (écoordonnes d'origine)
G1 F10 X0 G4 P0.1
G91 G81 X1 Y0 Z-1 R1 L4 (cycle de çperage)
G90 G0 X0 Y1
Z0
G91 G81 X1 Y0 Z-.5 R1 L4 (cycle de çperage)
G80 (érvocation du cycle G81)
M2 (fin de programme)
```




*Exemple 6:* Douze trous en carré

Cet exemple montre l'utilisation du mot L pour répéter une série incrémentale de cycles de perçage pour des blocs de code successifs dans le même mode mouvements G81. Ici, nous produisons 12 trous au moyen de cinq lignes de code dans le mouvement modal.

```
G90 G0 X0 Y0 Z0 (écoordonnes d'origine)
G1 F50 X0 G4 P0.1
G91 G81 X1 Y0 Z-0.5 R1 L4 (cycle de çperage)
X0 Y1 R0 L3 (éérptition)
X-1 Y0 L3 (éérptition)
X0 Y-1 L2 (éérptition)
G80 (érvocation du cycle G81)
G90 G0 X0 (retour vers l'origine en vitesse rapide)
Y0
Z0
M2 (fin de programme)
```




La deuxième raison d'utiliser les cycles de perçages, c'est qu'il produisent un mouvement préliminaire et retournent à une position prévisible et contrôlable, quel que soit le point de départ du cycle.

## 15.47 G90, G91: Modes de déplacement

– *G90* est le mode de déplacement absolu, les valeurs d'axes *X, Y, Z, A, B, C, U, V, W* représentent les positions dans le système de coordonnées courant. Les exceptions à cette règle sont décrites dans la section sur les cycles de perçage.
– *G91* est le mode de déplacement relatif, en mode relatif, les valeurs d'axes représentent un incrément depuis la position courante.

**Exemple avec G90**

```
G90 (passe en mode de édplacement absolu)
G0 X2.5 (édplacement élinaire en vitesse rapide à la écoordonne X=2.5 en
incluant tous les offsets en cours)
```

**Exemple avec G91**

```
G91 (passe en mode de édplacement relatif)
G0 X2.5 (édplacement élinaire en vitesse rapide, à +2.5 en X de la position
courante)
```

– Voir G0 pour plus d'information.

## 15.48 G90.1, G91.1: Mode de déplacement en arc (I, J et K)

– *G90.1* - Mode de déplacement absolu pour les offsets I, J et K. Quand G90.1 est actif, I et J doivent être tous les deux spécifiés avec G2/G3 pour le plan XY ou J et K pour le plan XZ, sinon c'est une erreur.
– *G91.1* - Mode de déplacement relatif pour les offsets I, J et K. G91.1 replace I, J et K à leur fonctionnement normal.

## 15.49 G92 Décalage d'origine des systèmes de coordonnées

```
G92 axes
```

Voir ce chapitre pour une vision générale des systèmes de coordonnées.

G92 fixera de nouvelles valeurs de coordonnées au point actuel (sans faire de mouvement). Les mots d'axes contiennent les valeurs souhaitées. Au moins un mot d'axe est obligatoire, les autres sont facultatifs. Si il n'y a pas de mot d'axe pour un axe donné, les coordonnées de cet axe resteront inchangées.

Quand *G92* est exécuté, les origines de tous les systèmes de coordonnées sont déplacées. Elles seront déplacées de sorte que les valeurs du point contrôlé courant, dans le système de coordonnées courant, deviendront celles spécifiées dans la ligne du G92. Les origines de tous les systèmes de coordonnées sont décalées de la même distance.

Par exemple, supposons que le point courant soit à X=4 et qu'aucun décalage G92 ne soit actif. La ligne *G92 X7* est programmée, toutes les origines seront décalées de -3 en X, ce qui fera que le point courant deviendra X=7. Ce -3 est enregistré dans le paramètre 5211.

Être en mode de déplacement relatif est sans effet sur l'action de *G92*.

Des décalages G92 peuvent déjà être actifs quand *G92* est appelé. Si c'est le cas, ils seront remplacés par le nouveau décalage, de sorte que le point courant devienne la valeur spécifiée.

C'est une erreur si:
– Tous les mots d'axes sont omis.
LinuxCNC conserve les décalages G92 et les réutilise au prochain démarrage du logiciel. Pour éviter cela, programmer un *G92.1* qui les effacera, ou un G92.2 qui supprimera les valeurs enregistrées.

Voir le chapitre sur les systèmes de coordonnées.

Voir la section sur les décalages G92.

Voir la section sur les paramètres.

## 15.50 G92.1, G92.2 Remise à zéro des décalages des systèmes de coordonnées

– *G92.1* - Positionne les décalages d'axes à 0 et passe les paramètres 5211 à 5219 à zéro.
– *G92.2* - Positionne les décalages d'axes à 0, laisse les valeurs des paramètres inchangées, elles ne seront pas utilisées.

## 15.51 G92.3 Restauration des décalages d'axe

– *G92.3* - Positionne les décalages d'axes aux valeurs enregistrées dans les paramètres 5211 à 5219.

Il est possible de positionner les décalages d'axes dans un programme puis de ré-utiliser les mêmes dans un autre programme. Pour cela, programmer *G92* dans le premier programme, ce qui positionnera les paramètres 5211 à 5219. Ne pas utiliser G92.1 dans la suite du premier programme. Les valeurs des paramètres seront enregistrées lors de la sortie du premier programme et rétablies au chargement du second programme. Utiliser *G92.3* vers le début du deuxième programme, ce qui restaurera les décalages d'axes enregistrés par le premier.

## 15.52 G93, G94, G95: Choix des modes de vitesse

– *G93* - Passe en mode inverse du temps. Dans le mode vitesse inverse du temps, le mot *F* signifie que le mouvement doit être terminé en *[1/F]* minutes. Par exemple, si la valeur de *F* est *2.0*, les mouvements doivent être terminés en *1/2* minute.
  Quand le mode vitesse inverse du temps est actif, le mot *F* doit apparaître sur chaque ligne contenant un mouvement *G1*, *G2*, ou *G3*. Les mots F qui sont sur des lignes sans G1, G2, ou G3 sont ignorés. Être en mode vitesse inverse du temps est sans effet sur les mouvements G0 (vitesse rapide).
– *G94* - Passe en mode unités par minute. Dans le mode vitesse en unités par minute, le mot *F* indique le déplacement du point contrôlé en millimètres par minute, en pouces par minute, ou en degrés par minute, selon l'unité utilisée.
– *G95* - Passe en mode unités par tour. Dans le mode vitesse en unités par tour, le mot *F* donne le déplacement du point contrôlé à effectuer sur l'axe Z, en millimètres par tour de broche ou en pouces, selon l'unité utilisée.

C'est une erreur si:
– Le mode vitesse inverse du temps est actif et qu'une ligne avec G1, G2, ou G3 (explicitement ou implicitement) n'a pas de mot F.
– Une nouvelle vitesse n'a pas été spécifiée après un passage en G94 ou G95.

## 15.53 G96, G97: Modes de contrôle de la broche

```
G96 <D-> S- (vitesse de coupe constante)
G97         (mode tr/mn)
```

– *D-* - Vitesse de broche maximale en tours par minute.
– *S-* - Vitesse de coupe constante.
– *G96 D- S-* - Passe à une vitesse de coupe constante de *S* pieds par minute, si G20 est actif, ou *S* mètres par minute, si G21 est actif. D- est facultatif.
  Lorsque *G96* est utilisé, s'assurer que *X0* dans le système de coordonnées en cours (y compris les compensations d'outils) est bien le centre de rotation, sinon LinuxCNC ne donnera pas la vitesse de broche désirée. *G96* n'est pas affecté par les mode rayon ou diamètre.
– *G97* - Vitesse de coupe en tr/mn.

**Exemple avec G96**

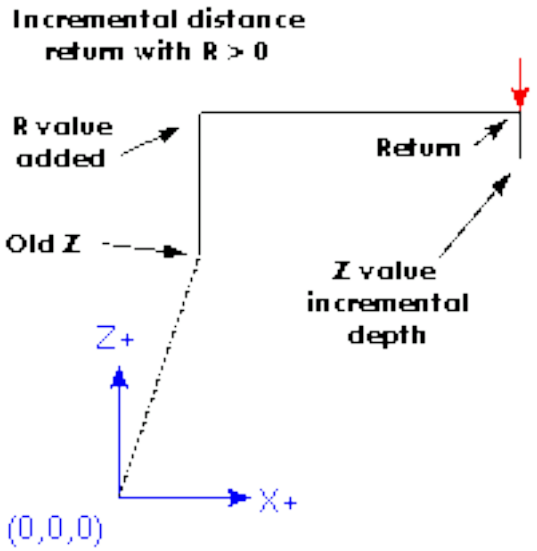
```
G96 D2500 S250 (passe à une vitesse de coupe constante de 250 m/mn maximum pour
une vitesse de broche maximale de 2500tr/mn).
```

C'est une erreur si:
– S n'est pas spécifié avec G96.
– Une vitesse est spécifiée en mode G96 et la broche ne tourne pas.

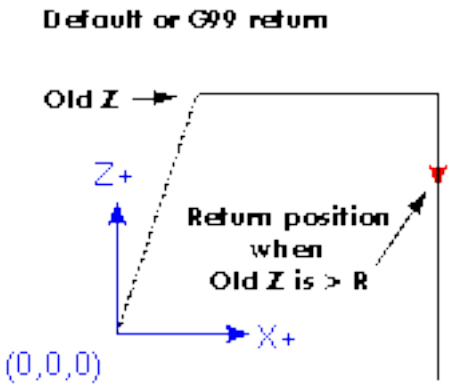
## 15.54 G98, G99: Options du plan de retrait

Quand la broche se rétracte pendant les cycles de perçage, il existe deux options pour indiquer comment elle doit se rétracter:

1. *G98* Retrait perpendiculaire au plan de travail courant jusqu'à la position qui était celle de cet axe juste avant le début du cycle de perçage. (à moins que cette position ne soit inférieure à celle indiquée par le mot R, auquel cas, c'est cette dernière qui serait utilisée).




1. *G99* Retrait perpendiculaire au plan de travail courant jusqu'à la position indiquée par le mot *R*.




Ne pas oublier que la signification du mot R change selon que le mode de déplacement est absolu ou relatif.

Le plan de retrait initial (G98) est annulé chaque fois que le mode de mouvement est abandonné, que ce soit explicitement avec G80 ou implicitement (tout code de mouvement qui n'est pas un cycle). Basculer d'un mode de cycle à un autre, par exemple entre G8[illegible]83 n'annule pas le plan de retrait initial. Il est permis de basculer entre G98 et G99 durant une série de cycles de perçage.

# Chapitre 16

# Les M-codes

## 16.1 Table des M-codes

| Code | Description |
|---|---|
| M0 M1 | Pause dans le programme |
| M2 M30 | Fin de programme |
| M60 | Pause pour déchargement pièce |
| M3 M4 M5 | Contrôle de la broche |
| M6 Tn | Appel d'outil n=numéro d'outil |
| M7 M8 M9 | Contrôle des arrosages |
| M19 | Orientation de la broche |
| M48 M49 | Contrôle des correcteurs de vitesse |
| M50 | Contrôle du correcteur de vitesse travail |
| M51 | Contrôle du correcteur de vitesse de broche |
| M52 | Correcteur dynamique de vitesse d'avance |
| M53 | Contrôle de la coupure de vitesse |
| M61 | Correction du numéro de l'outil courant |
| M62 à M65 | Contrôle de sortie numérique |
| M66 | Contrôle d'entrée numérique et analogique |
| M67 | Contrôle sortie analogique synchronisée |
| M68 | Contrôle sortie analogique directe |
| M70 | Enregistre l'état modal |
| M71 | Invalide l'état modal enregistré |
| M72 | Restaure l'état modal |
| M73 | Enregistrement/auto-restauration de l'état modal |
| M100 à M199 | M-codes définis par l'utilisateur |

## 16.2 M0, M1, pause dans le programme

– *M0* - Effectue une pause temporaire dans le programme en cours (quelle que soit la position du bouton d'arrêt facultatif). LinuxCNC reste en mode automatique afin que le MDI ou d'autres actions manuelles ne puissent pas être activés. Presser le départ cycle après cette commande relance le programme à la ligne suivante.
– *M1* - Stoppe temporairement le programme en cours (mais seulement si le bouton d'arrêt optionnel est activé). LinuxCNC reste en mode automatique afin que le MDI ou d'autres actions manuelles ne puissent pas être activés. Presser le départ cycle après cette commande relance le programme à la ligne suivante.

**Note**
Il est permis de programmer *M0* et *M1* en mode données manuelles (MDI), mais l'effet ne sera probablement pas perceptible, puisque le comportement normal en mode MDI est de s'arrêter, de toute façon, à la fin de chaque ligne.

## 16.3 M2, M30, fin de programme

– *M2* - Indique la fin du programme. Presser le départ cycle après cette commande relance le programme au début du fichier.
– *M30* - Décharge le porte-pièce du chargeur et termine le programme. Presser le départ cycle après cette commande relance le programme au début du fichier.

Les deux commandes précédentes produisent les effets suivants:

1. Changement du mode automatique au mode MDI.
2. Les décalages d'axes sont mis aux valeurs par défaut (comme avec *G54*).
3. Le plan de travail actif devient XY (comme avec *G17*).
4. Le mode de déplacement devient absolu (comme avec *G90*).
5. La vitesse travail passe en unités par minute (comme avec *G94*).
6. Les correcteurs de vitesse sont activés (comme avec *M48*).
7. Les compensations d'outil sont désactivées (comme avec *G40*).
8. La broche est arrêtée (comme avec *M5*).
9. Le mode mouvement courant devient *G1* (comme avec *G1*).
10. L'arrosage est arrêté (comme avec *M9*).

**Note**
Les lignes de code placées après un *M2* ou un *M30* ne sont pas exécutées.

## 16.4 M60, pause pour déchargement pièce

– *M60* - Procède au changement de porte-pièce avec le chargeur de pièces et effectue une pause dans le programme en cours (quel que soit le réglage du bouton d'arrêt facultatif). Presser ensuite le bouton de départ cycle pour relancer le programme à la ligne suivante.

## 16.5 M3, M4, M5 Contrôle de la broche

– *M3 Snnnnn* - Démarre la broche en sens horaire à la vitesse **nnnnn**.
– *M4 Snnnnn* - Démarre la broche en sens anti-horaire à la vitesse **nnnnn**.
– *M5* - Arrête la rotation de la broche.

Il est permis d'utiliser *M3* ou *M4* si la vitesse de broche est à zéro. Si cela est fait (ou si le bouton du correcteur de vitesse est activé mais mis à zéro), la broche ne tournera pas. Si, plus tard la vitesse de broche est augmentée (ou que le correcteur de vitesse est augmenté), la broche va se mettre en rotation. Il est permis d'utiliser *M3* ou *M4* quand la broche est déjà en rotation ou d'utiliser *M5* quand la broche est déjà arrêtée.

## 16.6 M6 Appel d'outil

### 16.6.1 Changement d'outil manuel

Si le composant de HAL, hal_manualtoolchange est chargé, *M6* va arrêter la broche et inviter l'utilisateur à changer l'outil. Pour plus d'informations sur hal_manualtoolchange voir la section sur le changement manuel d'outil.

### 16.6.2 Changement d'outil

Pour changer l'outil, actuellement dans la broche, par un autre, nouvellement sélectionné en utilisant le mot T, voir la section sur le choix de l'outil, programmer *M6*. Un changement d'outil complet donnera:
- La rotation de la broche est arrêtée.
- L'outil qui a été sélectionné (par le mot T sur la même ligne ou sur n'importe quelle ligne après le changement d'outil précédent), sera placé dans la broche. Le mot **T** est un nombre entier indiquant le numéro de poche d'outil dans le carrousel (non son index).
- Si l'outil sélectionné n'est pas déjà dans la broche avant le changement d'outil, l'outil qui était dans la broche (s'il y en avait un) va être replacé dans son emplacement dans le chargeur.
- Les coordonnées des axes seront arrêtées dans les mêmes positions absolues qu'elles avaient avant le changement d'outil (mais la broche devra peut-être être réorientée).
- Aucune autre modification ne sera apportée. Par exemple, l'arrosage continue à couler durant le changement d'outil à moins qu'il ne soit arrêté par *M9*.

**AVERTISSEMENT**
La longueur d'outil n'est pas modifié par *M6*, utilisez un *G43* après le *M6* pour changer la longueur d'outil.

Le changement d'outil peut inclure des mouvements d'axes pendant son exécution. Il est permis (mais pas utile) de programmer un changement d'outil avec le même outil que celui qui est déjà dans la broche. Il est permis également, si il n'y a pas d'outil dans le slot sélectionné, dans ce cas, la broche sera vide après le changement d'outil. Si le slot zéro a été le dernier sélectionné, il n'y aura pas d'outil dans la broche après le changement.

## 16.7 M7, M8, M9 Contrôle de l'arrosage

- *M7* - Active l'arrosage par gouttelettes.
- *M8* - Active l'arrosage fluide.
- *M9* - Arrête tous les arrosages.

Il est toujours permis d'utiliser une de ces commandes, que les arrosages soient arrêtés ou non.

## 16.8 M19 Orientation de la broche

- *M19 R- Q- [P-]*
- *R* - Position à atteindre à partir de 0, cette valeur doit être comprise entre 0 et 360 degrés.
- *Q* - Durée d'attente en secondes pour compléter l'orientation. Si *motion.spindle.is_oriented* n'est pas devenue vraie dans le temps imparti par Q, une erreur de timeout se produira.
- *P* - Direction de rotation vers la position cible.
  - *0* - rotation pour petit mouvement angulaire (défaut)
  - *1* - rotation toujours en sens horaire (même direction qu'avec M3)
  - *2* - rotation toujours en sens anti-horaire (même direction qu'avec M4)

M19 est révoqué par M3,M4 ou M5.

L'orientation de la broche nécessite un codeur de position avec index, indiquant la position de la broche ainsi que sa direction de rotation.

Paramètres de réglage de la section [RS274NGC].

ORIENT_OFFSET = 0 à 360 (offset fixe en degrés, ajouté au mot R de M19)

Broches de HAL
- *motion.spindle-orient-angle* (sortie float) Orientation souhaitée pour M19. Valeur du paramètre R de M19 plus la valeur du paramètre d'ini [RS274NGC]ORIENT_OFFSET.

M19 est une commande du groupe modal 7, comme M3, M4 et M5.
- *motion.spindle-orient-mode* (sortie s32) Mode de rotation de la broche souhaité. Reflète le mot P de M19, Défaut = 0
- *motion.spindle-orient* (sortie bit) Indique le début du cycle d'orientation de la broche. Positionné par M19. Remis à zéro par M3,M4 ou M5. Si *spindle-orient-fault* n'est pas à zéro alors que *spindle-orient* est vraie la commande M19 échoue avec un message d'erreur.
- *motion.spindle-is-oriented* (entrée bit) Pin de confirmation de l'orientation de la broche. Termine le cycle d'orientation. Si *spindle-orient* est vraie quand *spindle-is-oriented* est activée, la pin *spindle-orient* est mise à zéro et la pin *spindle-locked* est activée. La pin *spindle-brake* est également activée.
- *motion.spindle-orient-fault* (entrée s32) Entrée de code d'erreur pour le cycle d'orientation. Toute valeur, autre que zéro, provoquera l'abandon du cycle d'orientation.
- *motion.spindle-locked* (sortie bit) Pin indiquant que le cycle de rotation est terminé. Désactivée par M3,M4 ou M5.

## 16.9 M48, M49 Contrôle des correcteurs de vitesse

- *M48* - Autorise les curseurs de corrections de vitesses de broche et celui de vitesse d'avance travail.
- *M49* - Inhibe les deux curseurs.

Il est permis d'autoriser ou d'inhiber ces curseurs quand ils sont déjà autorisés ou inhibés. Ils peuvent aussi être activés individuellement en utilisant les commandes *M50* et *M51*, voir ci-dessous.

## 16.10 M50 Contrôle du correcteur de vitesse travail

- *M50 <P1>* - Autorise le curseur de correction de vitesse d'avance travail. Le paramètre *P1* est optionnel.
- *M50 P0* - Inhibe le curseur de correction d'avance travail.

Quand il est inhibé, le curseur de correction de vitesse n'a plus aucune influence et les mouvements seront exécutés à la vitesse d'avance travail programmée. (à moins que ne soit actif un correcteur de vitesse adaptative).

## 16.11 M51 Contrôle du correcteur de vitesse broche

- *M51 <P1>* - Autorise le curseur de correction de vitesse de la broche. Le paramètre *P1* est optionnel.
- *M51 P0* - Inhibe le curseur de correction de vitesse de broche.

Quand il est inhibé, le curseur de correction de vitesse de broche n'a plus aucune influence, et la broche tournera à la vitesse programmée, en utilisant le mot *S* comme décrit dans la section sur le réglage de la vitesse de broche.

## 16.12 M52 Contrôle de vitesse adaptative

- *M52 P1* - Utilise une vitesse adaptative. Le paramètre *P1* est optionnel.
- *M52 P0* - Cesse l'utilisation d'une vitesse adaptative.

Quand la vitesse adaptative est utilisée, certaines valeurs externes sont utilisées avec les correcteurs de vitesse de l'interface utilisateur et les vitesses programmées pour obtenir la vitesse travail. Dans LinuxCNC, la HAL pin *motion.adaptive-feed* est utilisée dans ce but. Les valeurs de *motion.adaptive-feed* doivent être dans comprises entre 0 (vitesse nulle) et 1 (pleine vitesse).

## 16.13 M53 Contrôle de la coupure de vitesse

- *M53 P1* - Autorise le bouton de coupure de vitesse. Le paramètre *P1* est optionnel. Autoriser la coupure de vitesse permet d'interrompre les mouvements par le biais d'une coupure de vitesse. Dans LinuxCNC, la HAL pin *motion.feed-hold* est utilisée pour cette fonctionnalité. Une valeur de 1 provoque un arrêt des mouvements quand *M53* est actif.
- *M53 P0* - Inhibe le bouton de coupure de vitesse. L'état de *motion.feed-hold* est sans effet sur la vitesse quand *M53* est inhibé.

## 16.14 M61 Correction du numéro de l'outil courant

– 'M61 Q ' - Corrige le numéro de l'outil courant, en mode MDI ou après un changement manuel d'outil dans la fenêtre de données manuelles. Au démarrage de LinuxCNC avec un outil dans la broche, il est possible ainsi d'ajuster le numéro de l'outil courant sans faire de changement d'outil.

C'est une erreur si:
– Q n'est pas égal où supérieur à 0

## 16.15 M62 à M65 Contrôle de bits de sortie numérique

– *M62 P* - Active un bit de sortie numérique en synchronisme avec un mouvement.
– *M63 P* - Désactive un bit de sortie numérique en synchronisme avec un mouvement.
– *M64 P* - Active immédiatement un bit de sortie numérique.
– *M65 P* - Désactive immédiatement un bit de sortie numérique.

Le mot *P* spécifie le numéro du bit de sortie numérique. Le mot P doit être compris entre 0 et une valeur par défaut de 3. Si nécessaire, le nombre des entrées/sorties peut être augmenté en utilisant le paramètre *num_dio* lors du chargement du contrôleur de mouvement. Voir le manuel de l'intégrateur et section "LinuxCNC et HAL", pour plus d'informations.

Les commandes *M62* et *M63* seront mises en file d'attente. Toute nouvelle commande, destinée à un bit de sortie écrasera l'ancien réglage de ce bit. Plusieurs bits peuvent changer d'état simultanément par l'envoi de plusieurs commandes M62/M63.

Les nouveaux changements d'état des bits de sortie spécifiés, seront effectifs au début du prochain mouvement commandé. S'il n'y a pas de commande de mouvement ultérieur, les changements en attente n'auront pas lieu. Il est préférable de toujours programmer un G-code de mouvement (G0, G1, etc) juste après les M62/63.

*M64* et *M65* produisent leur effet immédiatement après être reçus par le contrôleur de mouvement. Ils ne sont pas synchronisés avec un mouvement.

**Note**
M62 à M66 ne seront opérationnels que si les pins *motion.digital-out-nn* appropriées sont connectées aux sorties dans le fichier HAL.

## 16.16 M66 Contrôle d'entrée numérique et analogique

```
M66 P- | E- <L-> <Q->
```

– *P* - Spécifie le numéro d'un bit d'entrée numérique entre 0 et 3.
– *E* - Spécifie le numéro d'un bit d'entrée analogique entre 0 et 3.
– *L* - Spécifie le mode d'attente.
  – Mode 0: *IMMEDIATE* - pas d'attente, retour immédiat, la valeur courante de l'entrée est stockée dans le paramètre #5399
  – Mode 1: *RISE* attente d'un front montant sur l'entrée.
  – Mode 2: *FALL* attente d'un front descendant sur l'entrée.
  – Mode 3: *HIGH* attente d'un état logique HAUT sur l'entrée.
  – Mode 4: *LOW* attente d'un état logique BAS sur l'entrée.
– *Q* - Spécifie le timeout pour l'attente, en secondes. Si le timeout est dépassé, l'attente est interrompue et la variable #5399 positionnée à -1.
– Le mode *0* est le seul autorisé pour une entrée analogique.

**Exemple de ligne avec M66**

```
M66 P0 L3 Q5 (attend jusquà' 5 secondes la émonte de l'éentre énumrique 0)
```

– *M66* attend un nouvel événement sur une entrée ou la fin de l'exécution du programme, jusqu'à ce que l'événement sélectionné (ou le timeout programmé) ne survienne. C'est également une erreur de programmer *M66* avec les deux mots, un mot P- et un mot E- (ce qui reviendrait à sélectionner à la fois une entrée analogique et une numérique).

Si nécessaire, le nombre des entrées/sorties peut être augmenté en utilisant les paramètres *num_dio* ou *num_aio* lors du chargement du contrôleur de mouvement. Voir le Manuel de l'intégrateur pour plus d'informations, section des configurations, paragraphes "LinuxCNC et HAL".

**Note**
M66 ne sera opérationnel que si les pins motion.digital-in-nn ou motion.analog-in-nn appropriées sont connectées aux entrées dans le fichier HAL.

## 16.17 M67 Contrôle de sortie analogique

```
M67 E- Q-
```

– *M67* - Contrôle une sortie analogique synchronisée avec un mouvement.
– *E* - Spécifie le numéro de la sortie, doit être compris entre 0 et 3.
– *Q* - Spécifie la valeur à appliquer sur la sortie.

Les changements de valeur spécifiés, seront effectifs au début du prochain mouvement commandé. S'il n'y a pas de commande de mouvement ultérieur, les changements en attente n'auront pas lieu. Il est préférable de toujours programmer un G-code de mouvement (G0, G1, etc) juste après les M67. M67 fonctionne comme M62 à M63.

Le nombre d'entrées/sorties peut être augmenté en utilisant le paramètre *num_aio* au chargement du contrôleur de mouvement. Voir les chapitres "LinuxCNC et HAL" dans la section configuration du Manuel de l'intégrateur pour plus d'informations sur le contrôleur de mouvement.

**Note**
M67 ne sera opérationnel que si les pins motion.analog-out-nn appropriées sont connectées aux sorties dans le fichier HAL.

## 16.18 M68 Contrôle de sortie analogique directe

```
M68 E- Q-
```

– *M68* - Contrôle directement une sortie analogique.
– *E* - Spécifie le numéro de la sortie, doit être compris entre 0 et 3.
– *Q* - Spécifie la valeur à appliquer sur la sortie.

M68 produit son effet immédiatement après être reçu par le contrôleur de mouvement. Il n'est pas synchronisé avec un mouvement. M68 fonctionne comme M64 à M65.

Le nombre d'entrées/sorties peut être augmenté en utilisant le paramètre *num_aio* au chargement du contrôleur de mouvement. Voir le chapitre "LinuxCNC et HAL" dans le Manuel de l'intégrateur pour plus d'informations sur le contrôleur de mouvement.

**Note**
M68 ne sera opérationnel que si les pins *motion.analog-out-nn* appropriées sont connectées aux sorties dans le fichier HAL.

## 16.19 M70 Enregistrement de l'état modal

Pour enregistrer explicitement l'état modal au niveau de l'appel courant, programmer *M70*. Une fois l'état modal enregistré avec *M70*, il peut être restauré exactement dans le même état en exécutant un *M72*.

Une paire d'instructions *M70* et *M72* est typiquement utilisée pour protéger un programme contre d'éventuels changements modaux pouvant se produire dans les sous-programmes.

Les états enregistrés sont les suivants:
- unités machine courantes G20/G21 (po/mm)
- plan de travail courant (G17/G18/G19 G17.1,G18.1,G19.1)
- statut de la compensation de rayon d'outil (G40,G41,G42,G41.1,G42,1)
- mode de déplacement - relatif/absolu (G90/G91)
- mode de vitesse (G93/G94,G95)
- coordonnées système courantes (G54-G59.3)
- statut de la compensation de longueur d'outil (G43,G43.1,G49)
- options du plan de retrait (G98,G99)
- mode de contrôle de broche (G96-css ou G97-RPM)
- mode de déplacement en arc (G90.1, G91.1)
- mode diamètre/rayon des tours (G7,G8)
- mode de contrôle de trajectoire (G61, G61.1, G64)
- avance et vitesse broche courantes (valeurs *F* et *S*)
- statut de la broche (M3,M4,M5) - on/off et direction
- statut de l'arrosage (M7) et (M8)
- réglages des correcteurs de vitesse broche (M51) et du correcteur de vitesse travail (M50)
- réglage du contrôle de vitesse adaptative (M52)
- réglage du contrôle de la coupure de vitesse (M53)

Noter qu'en particulier, les modes de mouvement (G1 etc) ne sont *PAS* restaurés.

*Le niveau de l'appel courant* signifie:
- Exécution dans le programme principal. Il n'y a qu'un seul emplacement de stockage pour l'état modal au niveau du programme principal; si plusieurs instructions *M70* sont exécutées tour à tour, seul l'état enregistré le plus récent est restauré quand un *M72* est exécuté.
- Exécution dans un sous-programme G-code. L'état enregistré par *M70* dans un sous-programme se comporte exactement comme un paramètre nommé local - on ne peut s'y référer qu'à l'intérieur du sous-programme en invoquant un *M72*, à la sortie du sous-programme, le paramètre disparaît.

Une invocation récursive d'un sous-programme introduit un nouveau niveau d'appel.

## 16.20 M71 Invalidation de l'état modal enregistré

L'état modal enregistré par M70 ou par M73 au niveau de l'appel courant est invalidé (ne peut plus être restauré nulle part).

Un appel ultérieur à *M72* sur le même niveau d'appel, échouera.

Si il est exécuté dans un sous-programme qui protège l'état modal par un *M73*, un *return* ou *endsub* ultérieur ne restaurera *PAS* l'état modal.

L'utilité de ce dispositif est douteuse. Il ne devrait pas être invoqué quand il peut disparaître.

## 16.21 M72 Restauration de l'état modal

L'état modal enregistré par un M70 peut être restauré en exécutant un *M72*.

La gestion de G20/G21 reçoit un traitement particulier car les avances sont interprétées différemment selon G20/G21: si les unités de longueur (mm/po) doivent être modifiées par une opération de restauration, *M72* va restaurer le mode distance en premier, puis ensuite tous les autres états, y compris les avances pour être sure que les valeurs d'avance soient interprétées selon un réglage d'unités correct.

C'est une erreur d'exécuter *M72* sans enregistrement précédent avec *M70* à ce niveau.

L'exemple suivant montre l'enregistrement puis la restauration de l'état modal autour de l'appel d'un sous-programme utilisant *M70* et *M72*. Noter que le sous-programme *imperialsub* n'est pas "au courant" des caractéristiques de M7x et peut être utilisé non modifié:

```
O<showstate> sub
(DEBUG, imperial=#<_imperial> absolute=#<_absolute> feed=#<_feed> rpm=#<_rpm>)
O<showstate> endsub

O<imperialsub> sub
g20 (imperial)
g91 (relative mode)
F5 (low feed)
S300 (low rpm)
(debug, in subroutine, state now:)
o<showstate> call
O<imperialsub> endsub

; programme principal
g21 (metric)
g90 (absolute)
f200 (fast speed)
S2500 (high rpm)

(debug, in main, state now:)
o<showstate> call

M70 (save caller state in at global level)
O<imperialsub> call
M72 (explicitely restore state)

(debug, back in main, state now:)
o<showstate> call
m2
```

## 16.22 M73 Enregistrement et auto-restauration de l'état modal

Pour enregistrer l'état modal à l'intérieur d'un sous-programme et restaurer cet état lors d'un *endsub* ou autre *return*, programmer *M73*.

En cas d'abandon d'un programme en cours d'exécution dans un sous-programme traitant un *M73*, l'état ne sera **PAS** restauré.

En outre, la fin normale (*M2*) d'un programme principal contenant un *M73* ne restaurera **pas** l'état.

L'utilisation suggérée consiste à placer au début d'un sous-programme, un O-code de sous-programme comme dans l'exemple ci-dessous. En utilisant *M73*, cette manière valide le design des sous-programmes qui doivent modifier l'état modal mais qui protège le programme appelant contre tout changement inopiné de l'état modal. Noter l'usage de paramètres nommés dans le sous-programme *showstate*.

```
O<showstate> sub
(DEBUG, imperial=#<_imperial> absolute=#<_absolute> feed=#<_feed> rpm=#<_rpm>)
O<showstate> endsub

O<imperialsub> sub
M73 (save caller state in current call context, restore on return or endsub)
g20 (imperial)
g91 (relative mode)
F5 (low feed)
S300 (low rpm)
(debug, in subroutine, state now:)
```

```
o<showstate> call

; note - M72 n'est pas éutilis ici - le endsub suivant ou un
; 'return' explicite restaurera lé'tat de l'appelant
O<imperialsub> endsub

; programme principal
g21 (metric)
g90 (absolute)
f200 (fast speed)
S2500 (high rpm)
(debug, in main, state now:)
o<showstate> call
o<imperialsub> call
(debug, back in main, state now:)
o<showstate> call
m2
```

## 16.23 Restauration sélective de l'état modal par le test de paramètres prédéfinis

Exécuter un *M72* ou au retour d'un sous-programme contenant un 'M73_ pour restaurer tout l'état modal enregistré.

Si seulement certains aspects de l'état modal doivent être préservés, une alternative consiste a utiliser les paramètres nommés prédéfinis, paramètres locaux et états conditionnels. L'idée est de rappeler les modes à restaurer au début du sous-programme et de restaurer ceux-ci avant de quitter. Voici un exemple, basé sur le programme *nc_files/tool-length-probe.ngc*:

```
O<measure> sub   (measure reference tool)
;
#<absolute> = #<_absolute>  (remember in local variable if G90 was set)
;
g30 (above switch)
g38.2 z0 f15 (measure)
g91 g0z.2 (off the switch)
#1000=#5063 (save reference tool length)
(print,reference length is #1000)
;
O<restore_abs> if [#<absolute>]
    g90 (restore G90 only if it was set on entry:)
O<restore_abs> endif
;
O<measure> endsub
```

## 16.24 M100 à M199 Commandes définies par l'utilisateur

```
M1-- <P- Q->
```

– *M1* -- - Un entier compris entre 100 et 199.
– *P* - Un nombre passé comme premier argument au programme externe.
– *Q* - Un nombre passé comme second argument au programme externe.

Le programme externe, nommé *M100* à *M199*, (avec un *M* majuscule et aucune extension) qui doit se trouver dans le répertoire pointé par la variable *[DISPLAY] PROGRAM_PREFIX* du fichier ini, sera exécuté avec les valeurs *P*- et *Q*- comme étant ses deux arguments. L'exécution du fichier G-code courant passera en pause jusqu'à ce que le programme invoqué soit terminé. Tout fichier exécutable valide peut être utilisé. Le fichier doit se trouver dans le chemin spécifié dans le fichier ini de configuration. Voir la section sur le fichier de configuration dans le manuel de l'intégrateur.

Après la création d'un nouveau programme M1nn, l'interface graphique doit être redémarrée pour que le nouveau programme soit pris en compte, autrement une erreur *M-code inconnu* surviendra.

**AVERTISSEMENT**
Ne pas utiliser un traitement de texte pour créer ou éditer ces fichiers. Un traitement de texte ajoute des caractères invisibles qui causent des problèmes et empêchent les scripts bash ou Python de fonctionner. Pour ces raisons, utiliser un éditeur de texte tel que *Gedit* dans Ubuntu ou le Notepad++ dans un autre OS.

Le message d'erreur *M-code inconnu* signifie que:
– La commande utilisateur spécifiée n'existe pas.
– Le fichier n'a pas été rendu exécutable.
– Le nom du fichier comporte une extension.
– Le nom du fichier ne suis pas le format suivant: M1nn où nn = 00 à 99.
– Le nom de fichier utilise un *m* minuscule.
Exemple d'utilisation, dans un programme G-code, on doit ouvrir et fermer un mandrin automatique via une broche du port parallèle, on appellera respectivement M101 pour ouvrir le mandrin et M102 pour le fermer. Les deux scripts bash correspondants, appelés M101 et M102 seront créés avant le lancement de LinuxCNC puis rendus exécutables, par exemple par un clic droit puis *propriétés* → *permissions* → *Exécution*. S'assurer que cette broche du port parallèle n'est pas déjà utilisée dans un fichier de HAL.

**Exemple de fichier pour M101**

```
#!/bin/bash
# ce fichier met la broche 14 du port à 1 pour ouvrir le mandrin automatique
halcmd setp parport.0.pin-14-out True
exit 0
```

**Exemple de fichier pour M102**

```
#!/bin/bash
# ce fichier met la broche 14 du port à 0 pour fermer le mandrin automatique
halcmd setp parport.0.pin-14-out False
exit 0
```

Pour passer des variables à un fichier M1nn, utiliser les mots facultatifs P et Q de cette façon:

```
M100 P123.456 Q321.654
```

**Exemple pour M100**

```
#!/bin/bash
tension=$1
vitesse=$2
halcmd setp thc.voltage $tension
halcmd setp thc.feedrate $vitesse
exit 0
```

Pour ouvrir un message graphique et passer en pause jusqu'à ce que la fenêtre du message soit fermée, utiliser un programme comme *Eye of Gnome* pour afficher le fichier graphique. Quand la fenêtre sera fermée, le programme reprendra.

**Exemple pour M110, affichage d'un graphique avec passage en pause**

```
#!/bin/bash
eog /home/robert/linuxcnc/nc_files/message.png
exit 0
```

Pour afficher un message graphique en continuant le traitement du fichier G-code, ajouter un caractère esperluette à la commande.

**Exemple pour M110, affichage d'un graphique sans passer en pause**

```
#!/bin/bash
eog /home/robert/linuxcnc/nc_files/message.png &
exit 0
```

# Chapitre 17

# Les O-codes

## 17.1 Utilisation des O-codes

Les O-codes permettent le contrôle de flux dans les programmes NGC. Ils commencent par une lettre **O**, qu'il ne faut pas confondre avec le chiffre **0**. Chaque bloc est associé à une adresse, qui est la valeur utilisée après la lettre **O**. Il faut prendre soin de bien faire correspondre les adresses des O-codes.

**Exemple de numérotation**

```
o100 sub
(noter que les blocs if - endif utilisent des énumros édiffrents)
  o110 if [#2 GT 5]
    (du code ici)
  o110 endif
  (encore du code ici)
o100 endsub
```

Le comportement est indéfini si:
- Le même nombre est utilisé pour plusieurs blocs
- D'autres mots sont utilisés sur une ligne contenant un mot O-.
- Un commentaire est utilisé sur une ligne contenant un mot O-.

**ASTUCE**
L'utilisation de la lettre **o** minuscule facilite la distinction avec le chiffre **0** qui peut être tapé par erreur. Par exemple: `o100` est plus facile à distinguer de `0100` que `O100`.

## 17.2 Sous-programmes: sub, endsub, return, call

Les sous-programmes s'étendent d'un *O- sub* à un *O- endsub*. Les lignes, à l'intérieur du sous-programme (le corps du sous-programme), ne sont pas exécutées dans l'ordre, mais elles sont exécutées à chaque fois que le sous-programme est appelé avec un *O-call*.

**Exemple de sous-programme**

```
O100 sub (sous-programme de mouvement rapide à l'origine)
  G53 X0 Y0 Z0
O100 endsub
  (autres lignes)
O100 call (ici, appel du sous-programme)
M2
```

Pour plus de détails sur ces instructions voir:
– mouvement G53,
– mouvement rapide G0,
– fin de programme M2.

**O- return** À l'intérieur d'un sous-programme, *O- return* peut être exécuté, pour retourner immédiatement au code appelant, comme si *O- endsub* avait été rencontré.

**Exemple avec O- return**

```
o100 sub
  o110 if [#2 GT 5] (teste si le èparamtre #2 est ésuprieur à 5)
    o100 return (si le test est vrai, retourne au édbut du sous-programme)
  o110 endif
  (autre code ici, qui sera ééexcut si le èparamtre #2 est éinfrieur à 5)
o100 endsub
```

Voir également les sections:
– les opérateurs binaires,
– les paramètres.

**O- call** *O- call* peut prendre jusqu'à 30 arguments optionnels, qui sont passés au sous-programme comme *#1*, *#2* , . . ., *#N*. Les paramètres de *#N+1* à *#30* ont la même valeur dans le contexte de l'appel. Au retour du sous-programme, les valeurs des paramètres #1 jusqu'à #30 (quel que soit le nombre d'arguments) sont restaurés aux valeurs qu'ils avaient avant l'appel.

Parce que *1 2 3* est analysé comme le nombre 123, les paramètres doivent être placés entre crochets. L'appel de sous-programme suivant, s'effectue avec 3 arguments:

**Exemple d'appel O-**

```
O200 call [1] [2] [3]
```

Les corps de sous-programme ne peuvent pas être imbriqués. Ils ne peuvent être appelés qu'après avoir été définis. Ils peuvent être appelés depuis d'autres fonctions et peuvent s'appeler eux même récursivement, s'il est judicieux de le faire. Le niveau maximum d'imbrication des sous-programmes est de 10.

Les sous-programmes n'ont pas de *valeur de retour*, mais ils peuvent changer la valeur des paramètres au dessus de #30 et ces changements sont visibles depuis le code appelant. Les sous-programmes peuvent aussi changer la valeur des paramètres nommés globaux.

## 17.3 Boucles: do, while, endwhile, break, continue

La boucle *while* a deux structures possibles: *while - endwhile* et *do - while*. Dans chaque cas, la boucle est quittée quand la condition du *while* devient fausse. La différence se trouve en fin de test de la condition. La boucle *do - while* exécute le code dans la boucle puis test la condition. La boucle *while - endwhile* effectue le test d'abord.

**Exemple avec while - endwhile**

```
(dessine la forme d'une dent de scie)
G0 X1 Y0 (édplacement en position de édpart)
#1 = 1 (assigne la valeur 1 au èparamtre #1)
F25 (fixe la vitesse d'avance travail)
o101 while [#1 LT 10]
  G1 X0
  G1 Y[#1/10] X1
  #1 = [#1+1] (éincrmente le compteur de test)
o101 endwhile
M2 (fin de programme)
```

**Exemple avec do - while**

```
#1 = 0 (assigne la valeur 0 au èparamtre #1)
o100 do
  (debug, èparamtre 1 = #1)
  o110 if [#1 EQ 2]
    #1 = 3 (assigne la valeur 3 au èparamtre #1)
    (msg, #1 s'est vu éassign la valeur 3)
    o100 continue (saute au édbut de la boucle)
  o110 endif
  (le code d'usinage ici)
  #1 = [#1 + 1] (éincrmente le compteur de test)
o100 while [#1 LT 3]
(msg, boucle étermine)
M2
```

À l'intérieur d'une boucle while, *O- break*, quitte immédiatement la boucle et *O- continue*, saute immédiatement à la prochaine évaluation de la condition du *while*. Si elle est vraie, la boucle recommence au début. Si elle est fausse, la boucle est quittée.

## 17.4 Conditionnel: if, elseif, else, endif

Le *if* conditionnel exécute un groupe d'instructions avec le même nombre *O* qui commence avec *if* et se termine avec *endif*. Les conditions optionnelles *elseif* et *else* peuvent se trouver entre le *if* et le *endif*.

Si la condition du *if* est vraie, les instructions qui suivent le *if* seront exécutées jusqu'à, au maximum, l'instruction conditionnelle suivante.

Si la condition du *if* est fausse, alors les instructions conditionnelles *elseif* suivantes seront évaluées l'une après l'autre. Si la condition du *elseif* est vraie alors les instructions suivant ce *elseif* seront exécutées jusqu'à l'instruction conditionnelle suivante. Si aucune des conditions du *if* ou du *elseif* n'est vraie, alors les instructions suivant le *else* seront exécutées. Quand une condition est vraie, les autres instructions conditionnelles du groupe ne sont plus évaluées.

**Exemple avec if - endif**

```
O102 if [#31 EQ 3] (si le èparamtre #31 est égal à 3 alors S2000)
    S2000
O102 endif
```

**Exemple avec if - elseif - else - endif**

```
o102 if [#2 GT 5] (si le èparamtre #2 est ésuprieur à 5 alors F100)
  F100
o102 elseif [#2 LT 2] (sinon si le èparamtre #2 est éinfrieur à 2 alors F200)
  F200
o102 else (sinon le èparamtre #2 vaut entre 2 et 5 alors F150)
  F150
o102 endif
```

## 17.5 Répétition: Repeat

La répétition *repeat*, exécutera les blocs contenus entre *repeat* et *endrepeat* le nombre de fois spécifié entre crochets. L'exemple suivant montre comment usiner une séries de 5 formes diagonales commençant à la position courante.

**Exemple avec repeat**

```
(Usine 5 formes diagonales)
G91 (Mode éincrmental)
O103 repeat [5]
    (éinsrer le code d'usinage ici)
```

```
    G0 X1 Y1 (Mouvement en diagonale vers la position suivante)
O103 endrepeat
G90 (Mode absolu)
```

## 17.6 Indirection

L'adresse de O- peut être donnée par un paramètre ou un calcul.

**Exemple d'indirection**

```
O[#101+2] call
```

**Calcul des valeurs dans les O-codes** Voici un condensé des sections utiles aux calculs des O-codes:

- les paramètres,
- les expressions,
- les opérateurs binaires,
- les fonctions.

## 17.7 Appel de fichier

Pour appeler un sous-programme par son nom, ce sous-programme doit contenir un *sub* et un *endsub*. Le fichier appelé doit se trouver dans le répertoire pointé par la variable *PROGRAM_PREFIX* ou *SUBROUTINE_PATH* du fichier ini. Les noms de fichiers ne peuvent inclure que des lettres **minuscules**, des chiffres, des points et des tirets bas. Un fichier de sous-programme nommé ne peut contenir qu'une seule définition de sous-programme.

**Exemple: l'appel d'un fichier nommé**

```
o<monfichier> call (appel un fichier énomm)
```

**Exemple: l'appel d'un fichier numéroté**

```
o123 call (appel un fichier éénumrot)
```

Dans le fichier appelé doit se touver le *sub* et le *endsub* correspondant à l'appel. Le fichier doit être un fichier valide.

**Exemple: le fichier monfichier.ngc appelé**

```
o<monfichier> sub
  (du code ici)
o<monfichier> endsub
M2
```

**Note**
Les noms de fichiers doivent être en lettres minuscules, ainsi *o<MonFichier>* sera transformé en *o<monfichier>* par l'interpréteur.

# Chapitre 18

# Les autres codes

## 18.1 F: Réglage de la vitesse d'avance travail

Pour régler la vitesse d'avance, programmer *F-*. L'application de la vitesse est telle que décrite dans l'aperçu global d'une machine numérique, section vitesse d'avance, à moins que le mode vitesse inverse du temps ne soit activé, dans ce cas, la vitesse est telle que décrite dans la section sur le choix des modes de vitesse.

## 18.2 S: Réglage de la vitesse de rotation de la broche

Pour régler la vitesse en tours par minute (tr/mn) de la broche, programmer *S-*. La broche va tourner à cette vitesse quand elle sera programmée pour tourner. Il est permis de programmer un mot *S* que la broche tourne ou non. Si le potentiomètre de correction de vitesse broche est autorisé et n'est pas positionné sur 100%, la vitesse de broche sera différente de celle programmée. Il est permis de programmer *S0*, la broche ne tournera pas.

C'est une erreur si:
– La valeur de S est négative.
Comme décrit dans la section sur le cycle de taraudage à droite, si un cycle de perçage *G84* (taraudage) est actif et que les potentiomètres de vitesse et d'avance sont autorisés, celui qui a le réglage le plus bas sera utilisé. La vitesse de rotation et d'avance resterons synchronisées. Dans ce cas, la vitesse peut différer de celle programmée, même si le potentiomètre de correction de vitesse travail est sur 100%.

## 18.3 T: Choix de l'outil

Pour sélectionner un outil, programmer *T-*, où la valeur de *T* correspond au numéro de la poche d'outil dans le carrousel. L'outil ne sera appelé et changé que quand un *M6* sera programmé voir la section sur l'appel d'outil. Le mot *T* peut apparaitre sur la même ligne que le *M6* ou sur une ligne précédente. Il est permis, mais normalement inutile, qu'un mot T apparaisse à plus de deux lignes avant, sans changement d'outil. Le carrousel peut bouger, seulement le plus récent mot T ne prendra effet qu'au prochain changement d'outil. Il est permis de programmer *T0*, aucun outil ne sera sélectionné. C'est utile pour avoir la broche vide.

C'est une erreur si:
– Un valeur négative de T est utilisée.
– Une valeur de T supérieure au nombre de poches d'outils dans le carrousel est utilisée.
Sur certaines machines, le carrousel se déplace lorsque le mot T est programmé, avec l'usinage en cours. Sur ces machines, programmer *Tn*, plusieurs lignes de texte avant le changement d'outil permet de gagner du temps. Une pratique de programmation courante pour ces types de machines, consiste à placer le mot T pour le prochain outil sur la ligne suivant le changement d'outil. Cela laisse au carrousel tout le temps pour se positionner.

Les mouvements rapides qui suivent un T<n> n'apparaissent pas sur l'écran de parcours d'outil d'Axis, et ce jusqu'au prochain mouvement en vitesse travail Cela se remarque surtout sur les machines ayant de longues distances de déplacement lors du changement d'outil, comme les tours. Cela peut prêter à confusion au début. Pour contourner ce dysfonctionnement pour l'outil courant, ajouter un G1 sans mouvement juste après le T<n>.

# Chapitre 19

# Exemples de fichiers G-Code

Après l'installation de LinuxCNC, plusieurs exemples de fichiers G-code se trouveront dans le répertoire /nc_files du dossier d'installation. Seuls les fichiers adaptés au type de la machine sont utilisable.

## 19.1 Exemples pour une fraiseuse

### 19.1.1 Fraisage hélicoïdal d'un orifice

– Nom du fichier: useful-subroutines.ngc
– Description: Programme pour usiner une poche ou un alésage avec utilisation des paramètres.

### 19.1.2 Rainurage

– Nom du fichier: useful-subroutines.ngc
– Decription: Programme pour fraiser une rainure avec utilisation des paramètres.

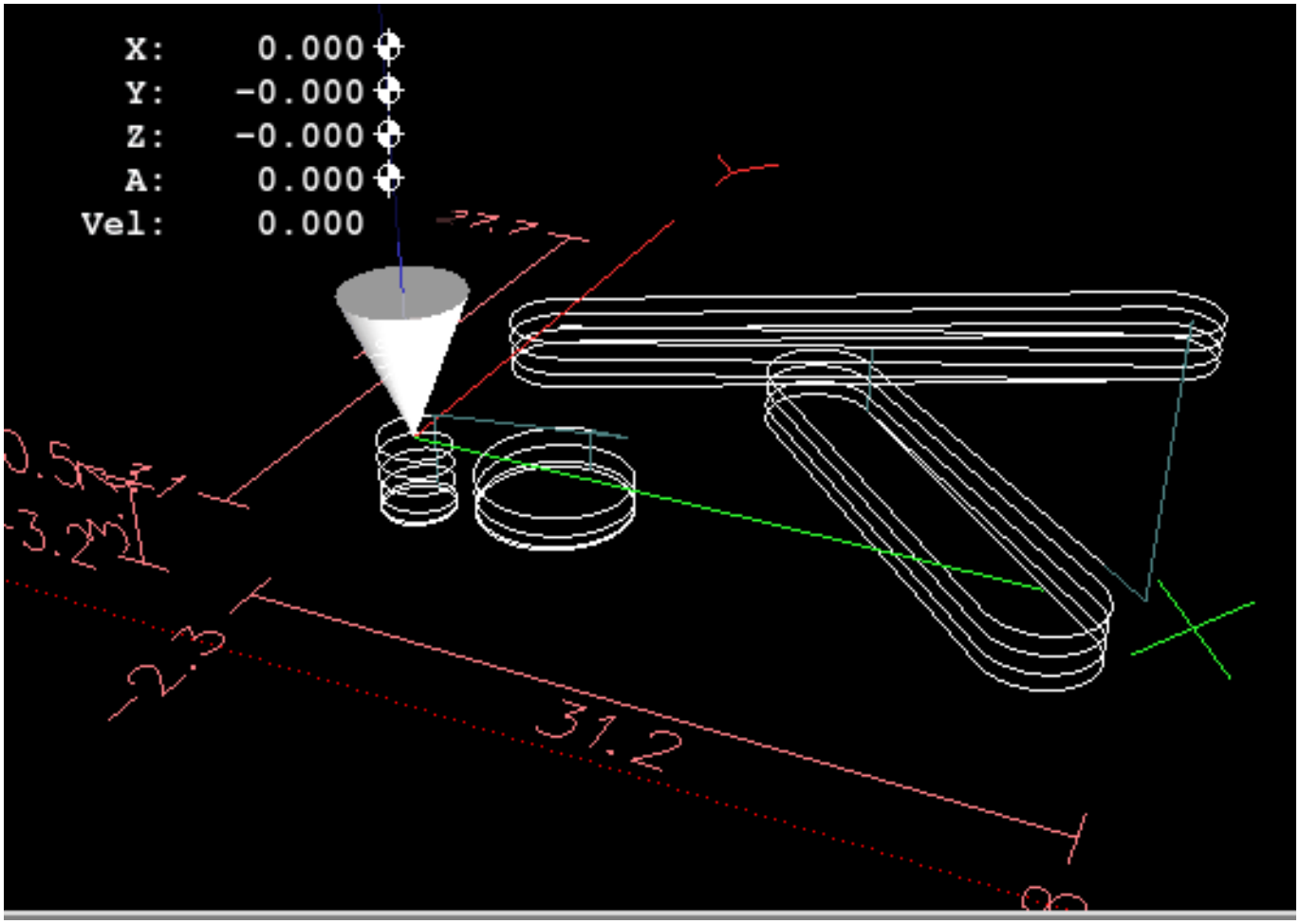



FIGURE 19.1 – Les différents usinages de l'exemple

### 19.1.3 Palpage d'une grille rectangulaire de points

– Nom du fichier: gridprobe.ngc
– Description: Relevé d'une grille rectangulaire de points au palpeur.

Ce programme palpe de manière répétitive, en suivant une grille régulière en XY et écrit les points mesurés dans le fichier *probe-results.txt* situé dans le même répertoire que le fichier .ini.

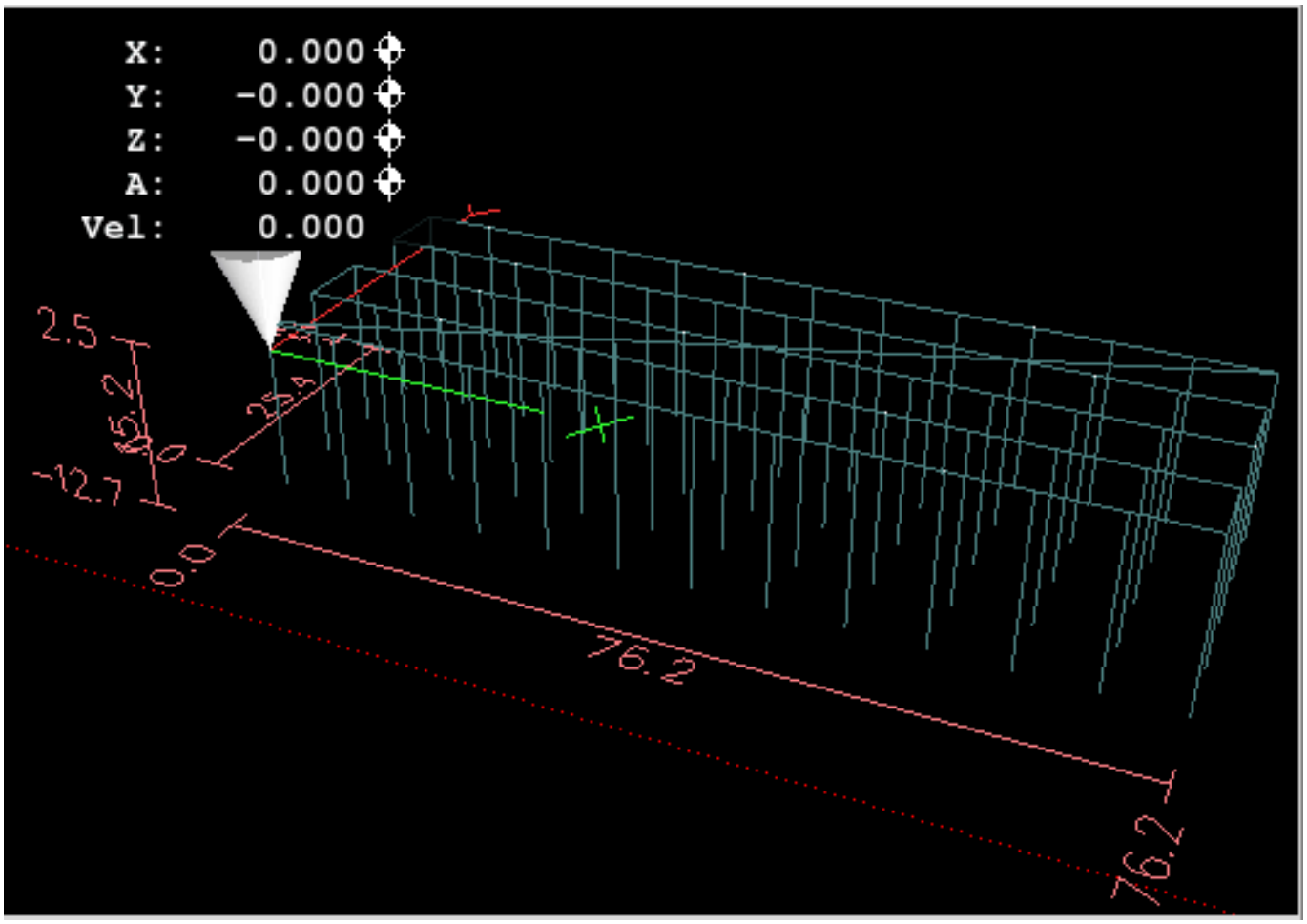



FIGURE 19.2 – La grille de palpage

### 19.1.4 Amélioration du palpage d'une grille rectangulaire de points

– Nom du fichier: smartprobe.ngc
– Description: Relevé d'une grille rectangulaire de points au palpeur.

Ce programme est une amélioration du précédent. Il palpe de manière répétitive, en suivant une grille régulière en XY et écrit les points mesurés dans le fichier *probe-results.txt* situé dans le même répertoire que le fichier .ini.

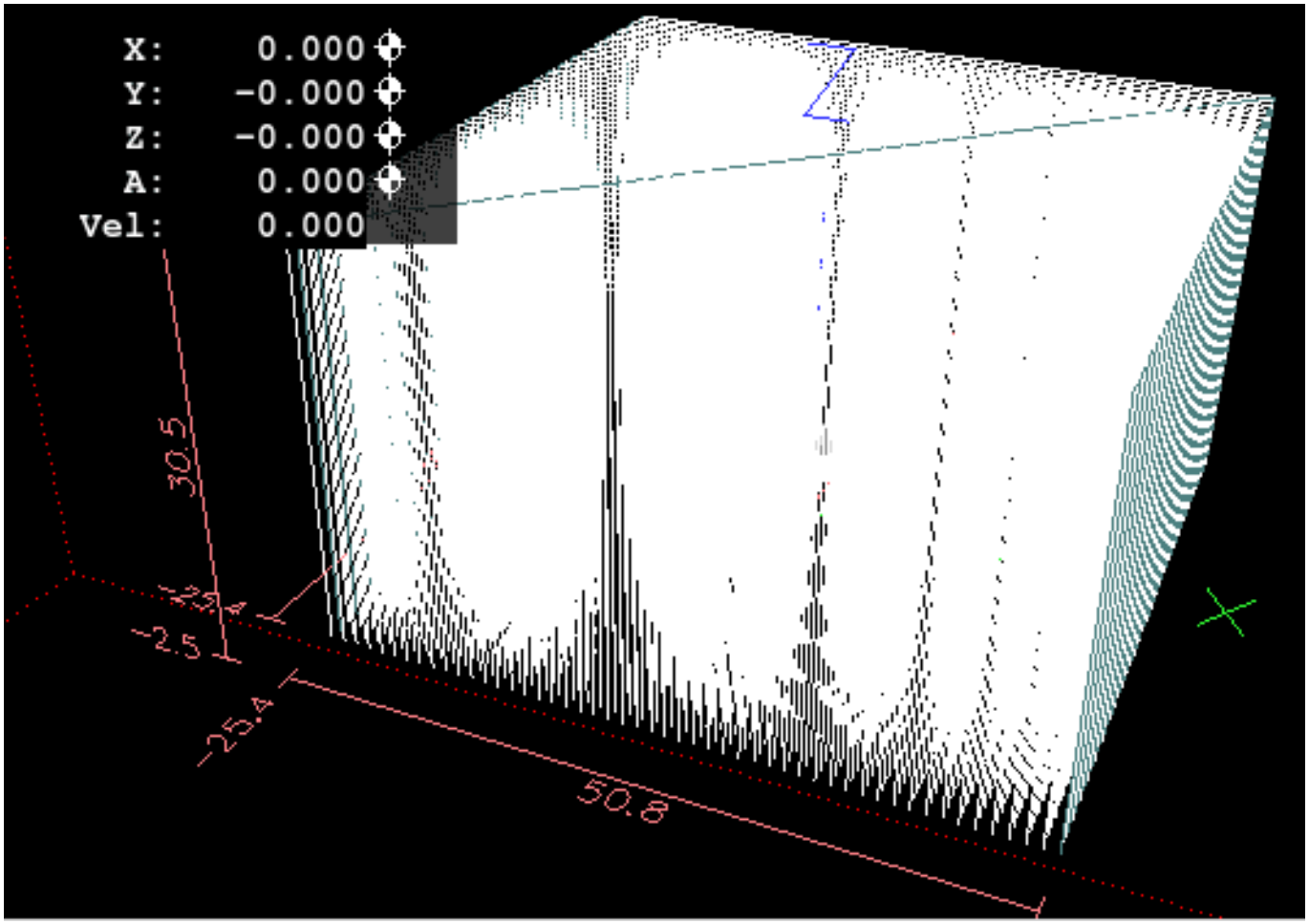



FIGURE 19.3 – La grille de palpage plus fine

### 19.1.5 Mesure de longueur d'outil

– Nom du fichier: tool-lenght-probe.ngc
– Description: Mesure automatique de la longueur de l'outil.

Ce programme donne un exemple de la mesure automatique de longueur d'outil en utilisant un contact raccordé à l'entrée sonde. C'est très pratique pour une machine sur laquelle la longueur des outils est différente à chaque montage.

### 19.1.6 Mesure d'un alésage au palpeur

– Nom du fichier: probe-hole.ngc
– Description: Mesure le centre et le diamètre d'un alésage.

Ce programme montre comment trouver le centre d'un alésage, comment calculer son diamètre et enregistrer les mesures dans un fichier.

### 19.1.7 Compensation de rayon d'outil

– Nom du fichier: comp-g1.ngc
– Description: Mouvements d'entrée et de sortie avec la compensation de rayon d'outil.

Ce programme démontre la particularité du chemin d'outil sans et avec la compensation de rayon d'outil. Le rayon d'outil est pris dans la table d'outils.

## 19.2 Exemples pour un tour

### 19.2.1 Filetage

– Nom du fichier: lathe-g76.ngc
– Description: Dressage, filetage et tronçonnage.

Ce programme donne un exemple de filetage automatique sur un tour avec utilisation des paramètres. Il demande quelques adaptations pour fonctionner, ces adaptations seront un excellent exercice.

# Chapitre 20

# Particularités des tours

Ce chapitre va regrouper les informations spécifiques aux tours, il est encore en cours de rédaction.

## 20.1 Mode tour

Si l'interface graphique Axis est utilisée et que la machine est un tour, pour qu'Axis représente les axes et la position de l'outil correctement il conviendra d'éditer le fichier *ini* et de modifier la section [DISPLAY] comme ceci:

```
[DISPLAY]

# Pour indiquer à Axis que la machine est un tour (lathe).
LATHE = TRUE
```

Le mode *tour* dans Axis ne fixe pas le plan par défaut comme étant G18 (XZ). Il est nécessaire de l'ajouter dans le préambule de tout programmer G-code ou, encore mieux, de l'ajouter directement dans le fichier *ini* comme cela:

```
[RS274NGC]

# G-code modaux pour initialiser l'éinterprteur
RS274NGC_STARTUP_CODE = G18 G20 G90
```

## 20.2 Fichier d'outils

La table d'outils est un fichier texte qui contient les informations de chaque outil. Ce fichier se trouve dans le même répertoire que le fichier *ini*, il est appelé *tool.tbl* par défaut. Les outils peuvent être dans un changeur d'outils ou simplement changés manuellement. Le fichier peut être édité avec un éditeur de texte ou être mis à jour en utilisant G10 L1, L10, L11. Axis peut aussi lancer l'éditeur de texte du système en y chargeant la table, l'opérateur peut ainsi intervenir directement sur les valeurs des outils. Le nombre maximum d'outils dans la table d'outils est de 56. Les numéros d'outil et de poches peuvent aller jusqu'à 99999.

Les versions antérieures de LinuxCNC avaient deux différents formats de table d'outils pour les fraiseuses et les tours, mais depuis la version 2.4.x, un format de table d'outil unique est utilisé. Il faut juste ignorer les parties de la table d'outils qui ne concernent pas la machine. Plus d'informations ici sur le format de la table d'outils.

## 20.3 Orientations des outils de tournage

La figure suivante montre les angles d'orientations des outils de tour ainsi que les informations sur l'angle frontal de l'arête de coupe (FRONTANGLE) et l'angle arrière de l'arête de coupe (BACKANGLE). Les positions vont croissantes dans le sens horaire par rapport à une ligne parallèle à l'axe Z, le zéro étant côté Z+.

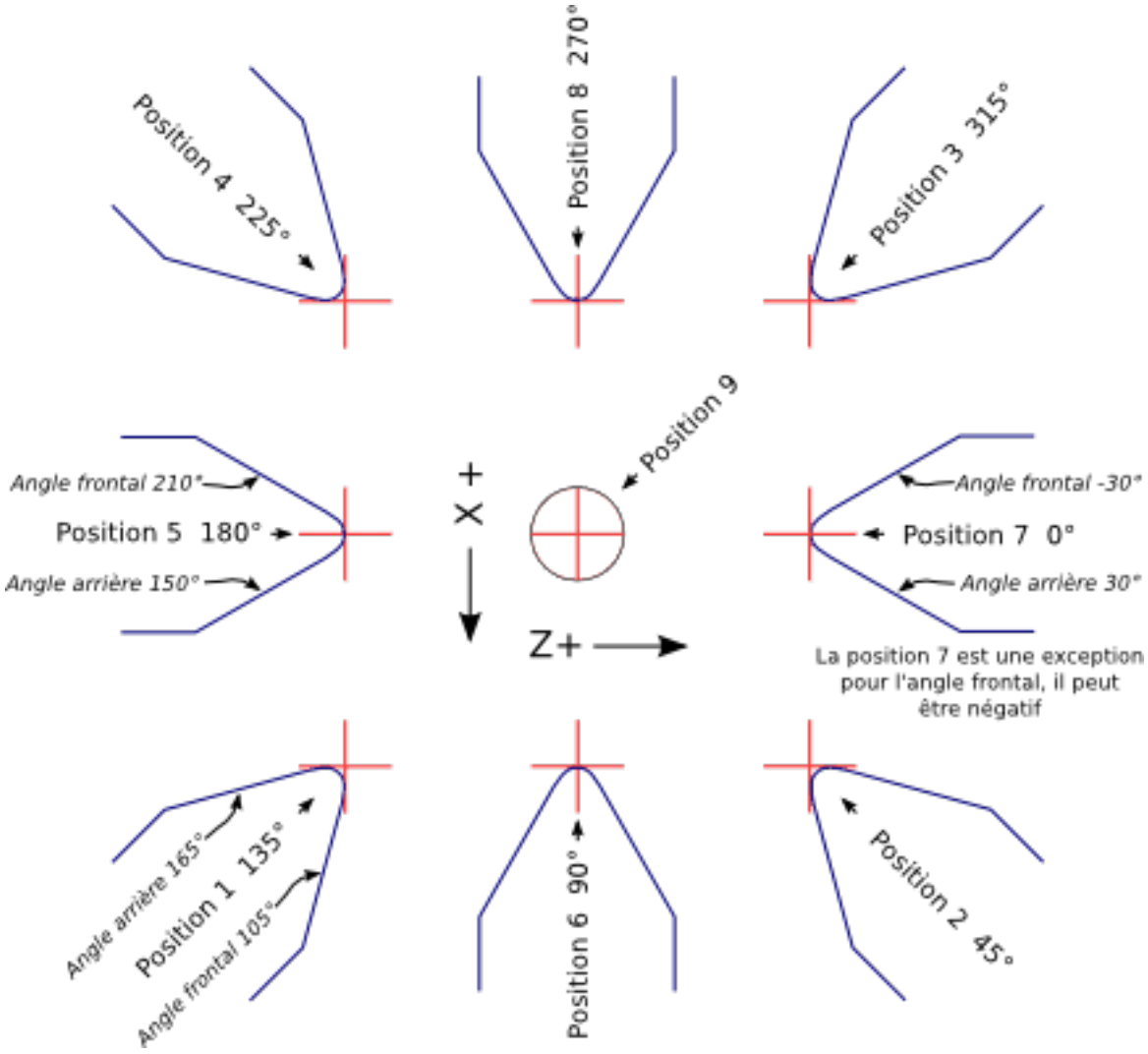
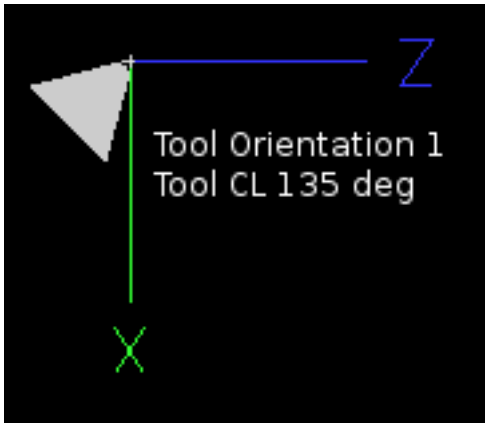
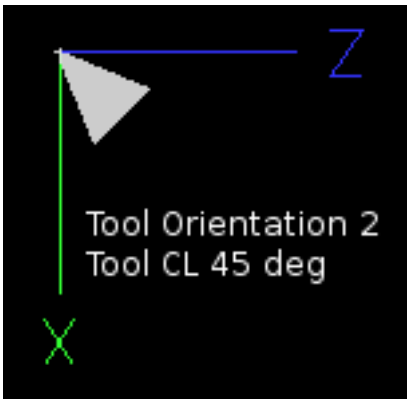


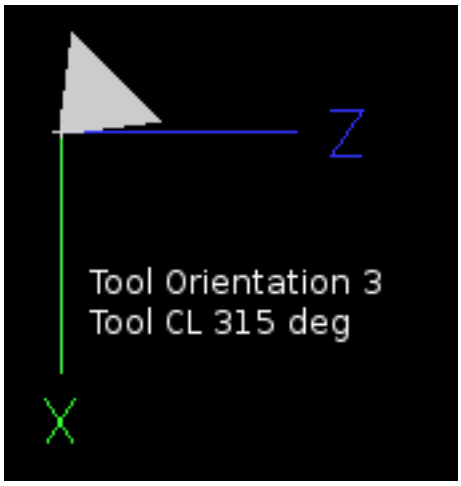

FIGURE 20.1 – Orientations des outils de tournage

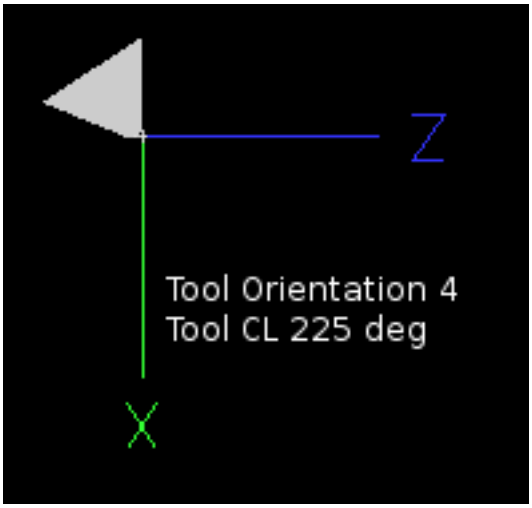
Les images ci-dessous montrent la représentation qu'Axis donne des orientations de l'outil, en se référant à la figure ci-dessus.

**Outil dans les positions 1, 2, 3 et 4**

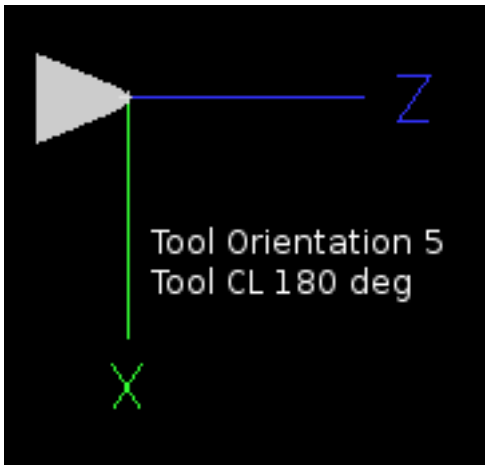


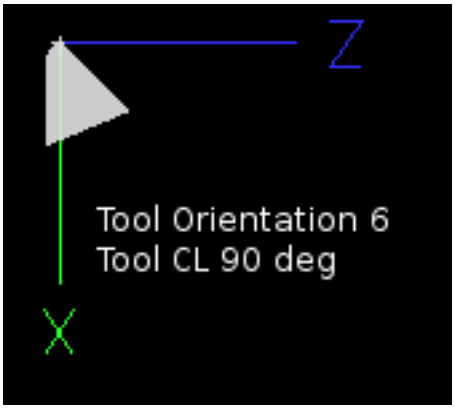


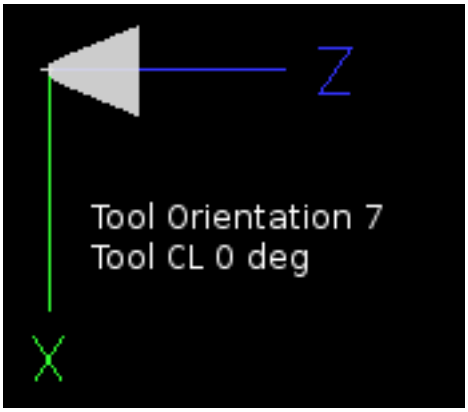



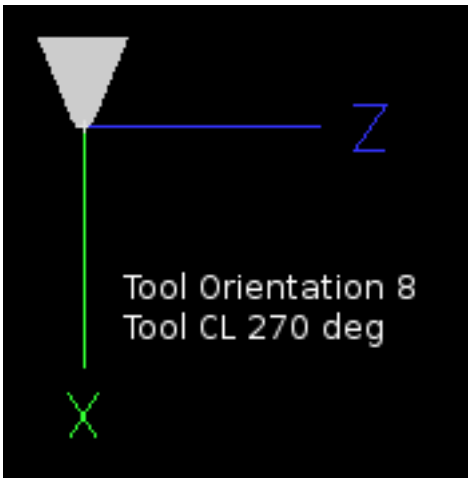




**Outil dans les positions 5, 6, 7 et 8**


Z
Tool Orientation 5
Tool CL 180 deg
X


Z
Tool Orientation 6
Tool CL 90 deg
X


Z
Tool Orientation 7
Tool CL 0 deg
X


Z
Tool Orientation 8
Tool CL 270 deg
X

## 20.4 Correction d'outil

Quand AXIS est utilisé sur un tour, il est possible de corriger l'outil sur les axes X et Z. Les corrections sont alors introduites directement dans la table d'outils en utilisant le bouton *Toucher* et sa fenêtre de dialogue.

### 20.4.1 Offset d'outil en X

L'offset X pour chaque outil correspond à un décalage de l'axe de la broche.

Une méthode consiste à prendre un outil de tournage standard et usiner un diamètre. Mesurer exactement ce diamètre puis, sans toucher à l'axe X, dans la fenêtre qui s'ouvre, après un appui sur le bouton *Toucher*, saisir le diamètre mesuré, ou le rayon si c'est le mode en cours. Ensuite, à l'aide d'encre à tracer ou d'un marqueur, recouvrir une zone sur la pièce, faire tangenter l'outil à cet endroit pour qu'il touche juste la surface encrée, ajuster alors l'offset X au diamètre mesuré de la pièce en utilisant le bouton *Toucher*. S'assurer que le rayon de bec est bien défini dans la table d'outils, pour que le point contrôlé soit correct. Le *Toucher* ajoute automatiquement un G43, de sorte que l'offset s'applique immédiatement à l'outil courant.

### 20.4.2 Séquence typique de correction d'outil en X:

1. Prise d'origine machine de chacun des axes, si ce n'est pas déjà fait.
2. Déclarer l'outil avec *M6 Tn* dans lequel *n* est le numéro de l'outil courant, présent en table d'outils.
3. Sélectionner l'axe X dans la fenêtre de l'onglet *Contrôle manuel (F3)*.
4. Déplacer l'axe X sur une position connue ou prendre une passe de test puis mesurer le diamètre obtenu.
5. Cliquer le bouton *Toucher* et choisir l'option *Table d'outils*, ce qui entrera la correction directement dans la table d'outil.
6. Recommencer la même séquence pour corriger l'axe Z.

Remarque: si le mode rayon est le mode courant, il faut évidemment entrer le rayon et non pas le diamètre.

### 20.4.3 Offset d'outil en Z

L'offset de l'axe Z peut être un peu déroutant au premier abord car il est composé de deux éléments. Le premier est l'offset de la table d'outils, le second est l'offset des coordonnées machine. Nous allons d'abord examiner l'offset de la table d'outils. Une méthode consiste à utiliser un point fixe sur le tour et à ajuster l'offset Z de tous les outils à partir de ce point fixe. Certains utilisent le nez de broche ou la face du mandrin. Cela donne la possibilité de changer d'outil et d'ajuster son offset Z, sans avoir à réinitialiser tous les outils.

### 20.4.4 Séquence typique de correction d'outil en Z:

1. Prise d'origine machine de tous les axes, si ce n'est pas déjà fait.
2. S'assurer qu'aucune compensation n'est activée pour le système de coordonnées courant.
3. Déclarer l'outil avec *M6 Tn* dans lequel *n* est le numéro de l'outil courant, présent en table d'outils.
4. Sélectionner l'axe Z dans la fenêtre de l'onglet *contrôle manuel (F3)*.
5. Placer un cylindre dans le mandrin.
6. Faire tangenter l'outil contre la face du cylindre.
7. Cliquer le bouton *Toucher* puis choisir *Table d'outils* et saisir la position à 0.0.
8. Répéter l'opération pour chaque outil, en utilisant le même cylindre.

Maintenant, tous les outils sont compensés à la même distance d'une position standard. Si un outil doit être changé, par exemple par un foret il suffira de répéter la séquence précédente pour qu'il soit synchronisé avec l'offset Z du reste des outils. Certains outils pourraient nécessiter un peu de réflexion pour déterminer le point contrôlé par rapport au point de *Toucher*. Par exemple, un outil de tronçonnage de 3.17mm d'épaisseur qui est touché sur le côté gauche, alors que l'opérateur veut Z0 sur le côté droit, il lui faudra alors saisir 3.17 dans la fenêtre du *Toucher*.

### 20.4.5 Machine avec tous les outils compensés

Une fois que tous les outils ont leurs offsets renseignés dans la table d'outils, il est possible d'utiliser n'importe quel outil présent en table d'outils pour ajuster le décalage du système de coordonnées machine.

### 20.4.6 Séquence typique de décalage du système de coordonnées:

1. Prise d'origine machine de tous les axes, si ce n'est pas déjà fait.
2. Déclarer l'outil avec *M6 Tn* dans lequel *n* est le numéro de l'outil courant, présent en table d'outils.
3. Envoyer un G43 pour que l'offset de l'outil soit activé. (voir ci-dessous)
4. Tangenter l'outil contre la pièce et fixer l'offset machine Z.

Ne pas oublier d'envoyer le G43 sur l'outil avant de définir le décalage du système de coordonnées machine, les résultats ne seraient pas ceux attendus... puisque la compensation de l'outil serait ajoutée à l'offset courant lorsque l'outil sera utilisé dans le programme.

## 20.5 Mouvements avec broche synchronisée

Sur un tour, les mouvements avec broche synchronisée nécessitent un signal de retour entre la broche et LinuxCNC. Généralement, c'est un codeur en quadrature qui fourni ce retour. Le manuel de l'intégrateur donne des explications sur l'utilisation des codeurs de broche.

**Filetage** Le cycle de filetage préprogrammé G76 est utilisé, tant en filetage intérieur qu'en filetage extérieur, voir la section G76.

**Vitesse de coupe à surface constante** La vitesse de coupe à surface constante utilise l'origine machine X modifiée par l'offset d'outil X, pour calculer la vitesse de rotation de la broche en tr/mn. La vitesse de coupe à surface constante permet de suivre les changements d'offset de l'outil. L'emplacement de l'origine machine de l'axe X doit être sur l'axe de rotation et doit se faire avec l'outil de référence (celui qui a l'offset à zéro).

**Avance par tour** L'avance par tour déplace l'axe Z de la valeur de F à chaque tour. Ce n'est pas destiné au filetage pour lequel il faut utiliser G76. D'autres informations sont dans la section sur G95.

## 20.6 Arcs

Le calcul des arcs peut être un exercice assez compliqué, même sur un tour, sans considérer les modes rayon et diamètre, ni l'orientation du système de coordonnées machine. Ce qui suit s'applique à des arcs au format centre. Sur un tour, il faut inclure G18 dans le préambule du programme G-code pour remplacer le G17 par défaut, le fait d'être en mode tour dans Axis ne suffit pas. Les arcs en G18, plan XZ utilisent les offsets pour I (l'axe X) et K (l'axe Z).

### 20.6.1 Les arcs et la cinématique du tour

Le tour classique a la broche à gauche de l'opérateur et l'outil entre l'opérateur et le centre de rotation du mandrin. C'est un agencement avec un axe Y(+) imaginaire pointant vers le sol.

Ce qui suit est valable pour ce type d'agencement:
- Le côté positif de l'axe Z pointe vers la droite, en s'éloignant de la broche.
- Le côté positif de l'axe X pointe vers l'opérateur, quand il est du côté de l'opérateur par rapport au centre de rotation, ses valeur sont positives.

Certains tours ont l'outil du côté arrière et un axe Y(+) imaginaire pointant vers le haut.

Les directions des arcs G2/G3 sont basées sur l'axe autour duquel ils tournent. Dans le cas des tours, il s'agit de l'axe imaginaire Y. Si l'axe Y(+) pointe vers le sol, il faut regarder vers le haut pour que l'arc paraisse aller dans la bonne direction. Alors qu'en regardant depuis le dessus il faut inverser les G2/G3 pour que l'arc semble aller dans la bonne direction.

### 20.6.2 Mode rayon et mode diamètre

Lors du calcul des arcs en mode rayon, il suffi de se rappeler la direction de rotation telle qu'elle s'applique à ce tour.

Lors du calcul des arcs en mode diamètre, X est le diamètre, l'offset X (I) est le rayon, même en mode diamètre G7.

## 20.7 Parcours d'outil

### 20.7.1 Point contrôlé

Le point contrôlé pour l'outil, suit la trajectoire programmée. Le point contrôlé est l'intersection entre deux lignes parallèles aux axes X et Z, tangentes au rayon de bec de l'outil, définies en faisant tangenter l'outil en X puis en Z. En cylindrage ou en dressage de face sur une pièce, la trajectoire de coupe et l'arête de coupe de l'outil suivent le même parcours. Lors du tournage d'un rayon ou d'un angle, l'arête de coupe de l'outil ne suit pas la trajectoire programmée, sauf si la compensation d'outil est activée. Dans la figure suivante, on voit bien que le point contrôlé n'est pas sur l'arête de coupe de l'outil comme on pourrait le supposer.

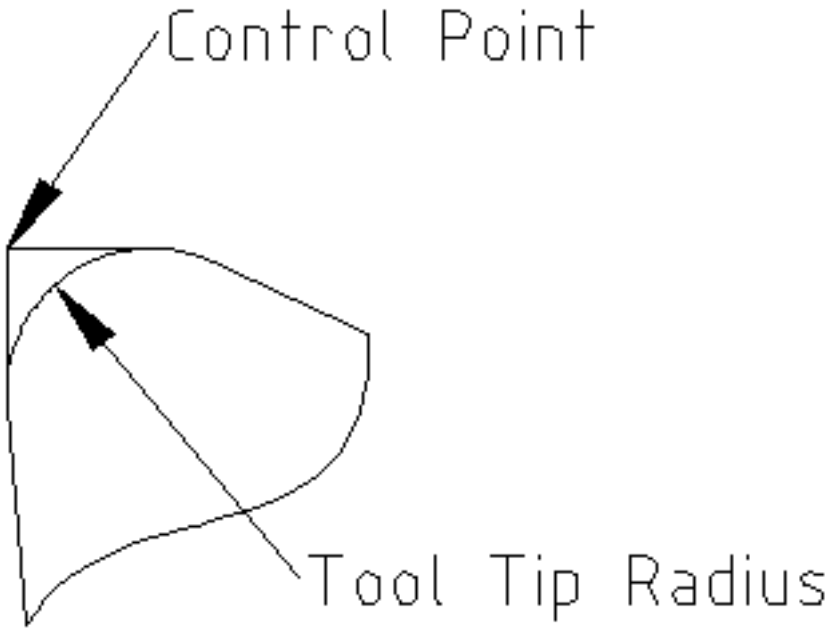



FIGURE 20.2 – Point contrôlé

### 20.7.2 Tourner les angles sans compensation d'outil

Maintenant imaginons de programmer une rampe sans compensation d'outil. La trajectoire programmée est représentée sur la figure suivante. Comme on peut le voir, la trajectoire programmée et la trajectoire de coupe souhaitée sont identiques uniquement si les mouvements de tournage suivent les axes X et Z.

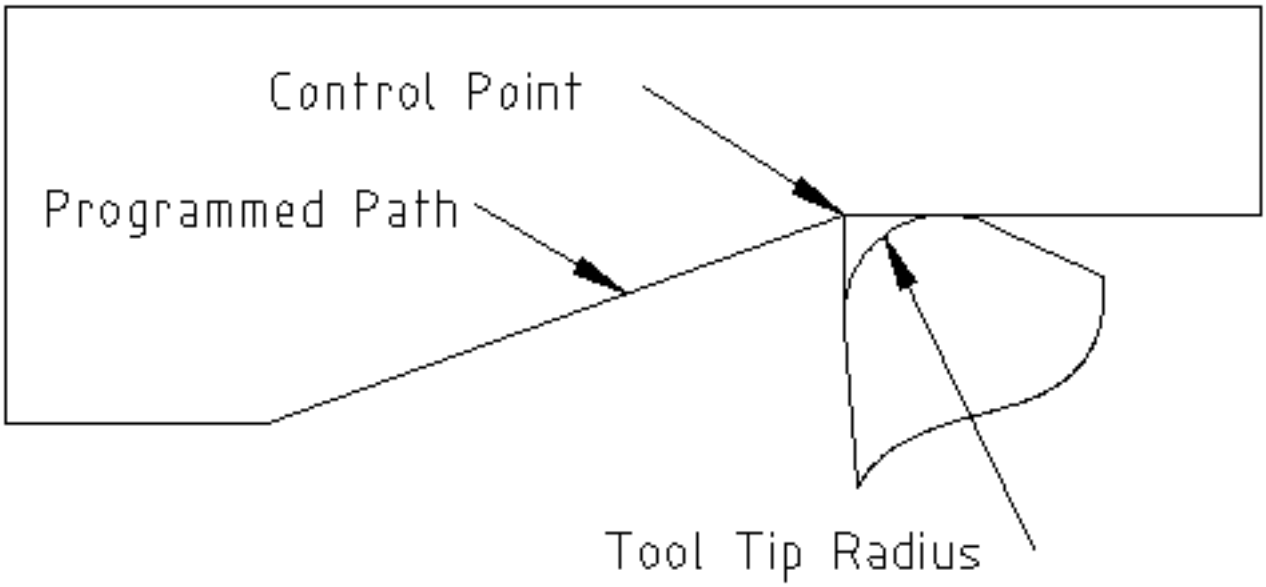



FIGURE 20.3 – Tournage en rampe

Le point contrôlé progresse en suivant la trajectoire programmée mais l'arête de coupe ne suit pas cette trajectoire comme c'est visible sur la figure suivante. Pour résoudre ce problème, il est nécessaire d'activer la compensation d'outil et d'ajuster la trajectoire programmée pour compenser le rayon de bec de l'outil.

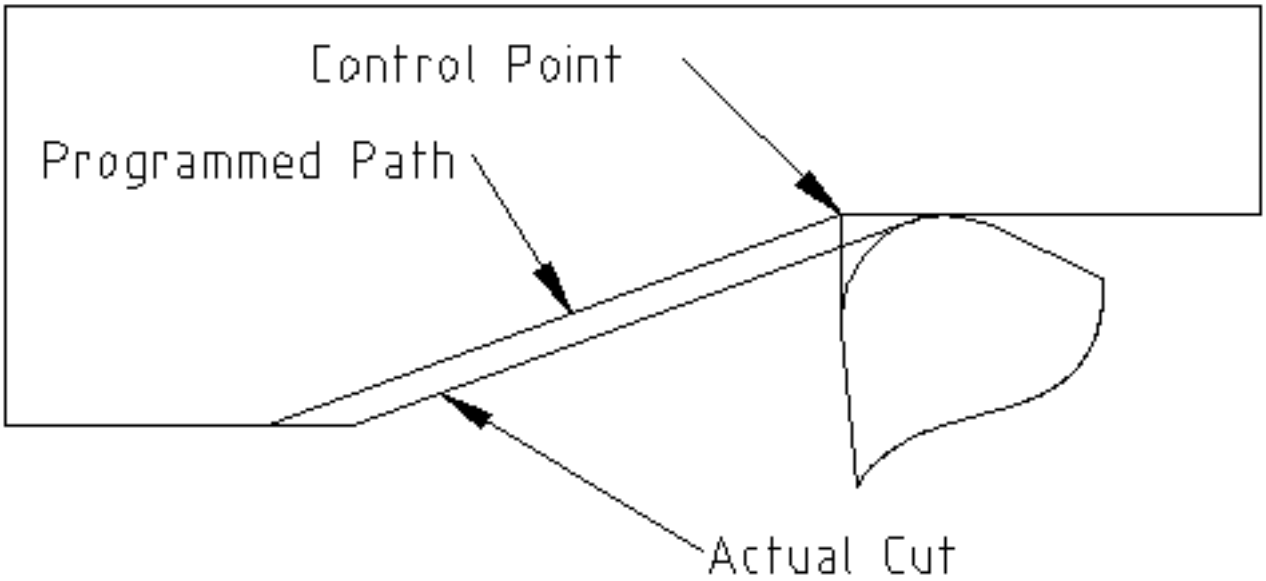



FIGURE 20.4 – Trajectoire en rampe

Dans l'exemple ci-dessus, pour suivre la rampe programmée et obtenir la bonne trajectoire, il suffi de décaler la trajectoire de la rampe vers la gauche, de la valeur d'un rayon de bec.

### 20.7.3 Tournage avec rayon extérieur

Dans cet exemple nous allons examiner ce qui se passe durant le tournage d'un rayon extérieur sans compensation de rayon de bec. Sur la figure suivante on voit l'outil tourner un diamètre extérieur sur la pièce. Le point contrôlé de l'outil suit bien la trajectoire programmée, l'outil touche le diamètre extérieur de la pièce.

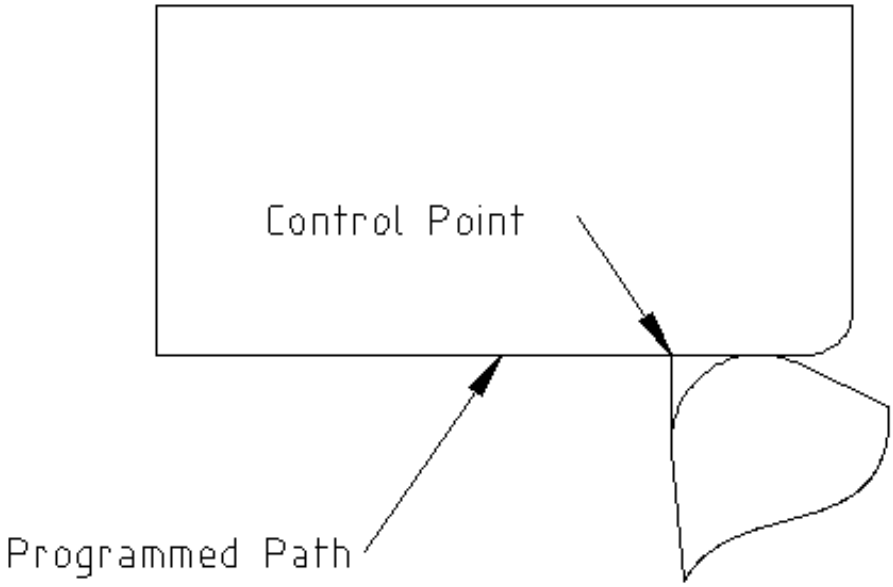



FIGURE 20.5 – Tournage du diamètre

Sur la figure suivante, on voit que quand l'outil approche la fin la pièce, le point contrôlé continue de suivre la trajectoire alors que l'arête de coupe a déjà quitté la matière et coupe en l'air. On voit aussi que malgré qu'un rayon a été programmé, la pièce conserve son angle d'extrémité.

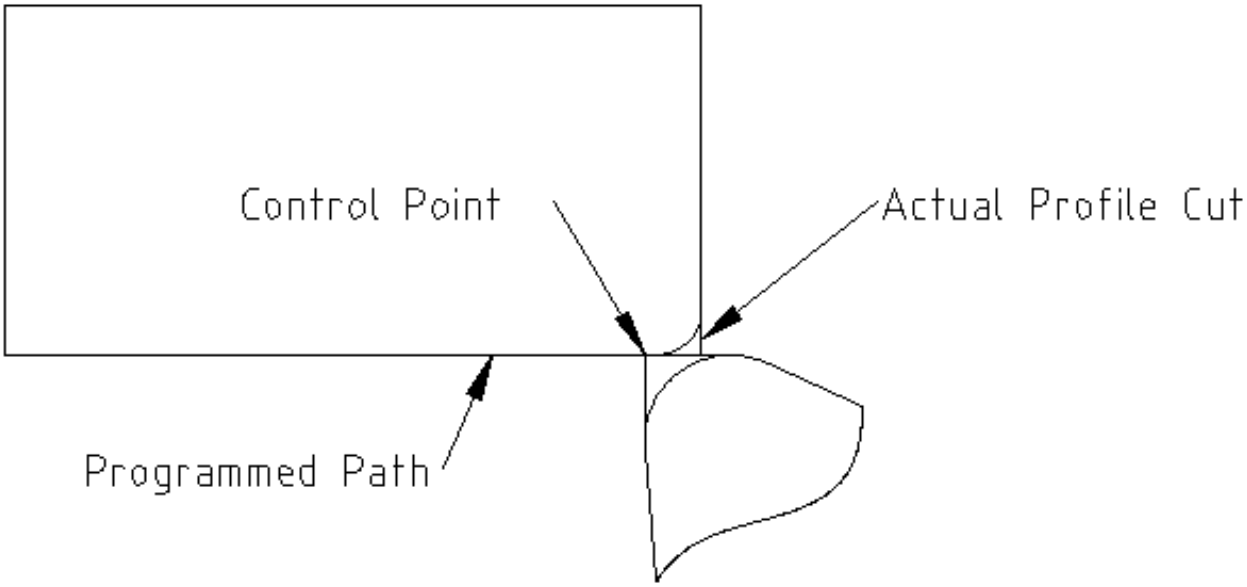



FIGURE 20.6 – Tournage du rayon

Maintenant, comme on le voit, le point contrôlé suit bien la trajectoire programmée mais l'arête de coupe est en dehors de la matière.

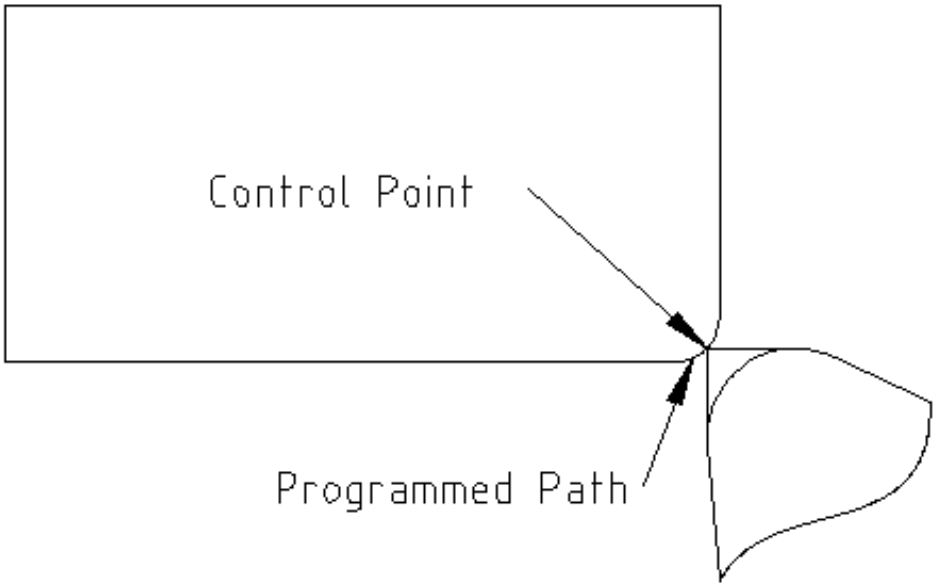



FIGURE 20.7 – Tournage du rayon

Sur la figure finale, on voit que l'arête de coupe a terminé le dressage de la face mais en laissant un coin carré à la place du beau rayon attendu. Noter aussi que, pour la même raison, pour ne pas laisser de téton au centre de la pièce lors du dressage de sa face, il convient de dépasser le centre de rotation de la valeur d'un rayon de bec de l'outil.

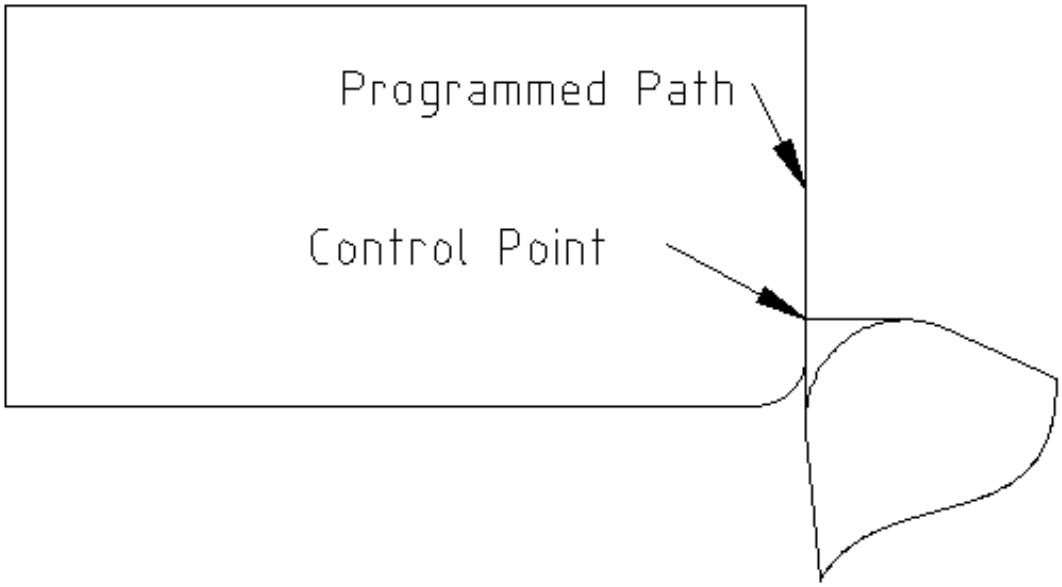



FIGURE 20.8 – Dressage de la face

### 20.7.4 Utiliser la compensation d'outil

- Quand la compensation d'outil est utilisée sur un tour, penser à l'arête de coupe de l'outil comme étant celle d'un outil rond.
- Quand la compensation d'outil est utilisée, la trajectoire doit être suffisamment large pour qu'un outil rond n'interfère pas avec la pièce à la ligne suivante.

- Pour tourner des lignes droites sur un tour, il est préférable de ne pas utiliser la compensation d'outil. Par exemple pour aléser un trou avec une barre d'alésage un peu grosse, la place pourrait manquer pour dégager l'outil et faire le mouvement de sortie.
- Le mouvement d'entrée dans un arc avec la compensation d'outil, est important pour obtenir des résultats corrects.

# Chapitre 21

# Différences avec RS274/NGC

## 21.1 Changements entre RS274/NGC et LinuxCNC

### 21.1.1 Position après un changement d'outil

Avec LinuxCNC, le mobile ne retourne pas sur la position de départ après un changement d'outil. Ce mode de fonctionnement est nécessaire, car un outil peut être plus long que l'outil précédent et dans ce cas un mouvement sur la position précédente placerait l'outil trop bas.

### 21.1.2 Les paramètres de décalage sont dans l'unité du fichier ini

Dans LinuxCNC, les valeurs mémorisées dans les paramètres pour les positions d'origine des commandes G28 et G30, les systèmes de coordonnées P1 à P9 et le décalage G92 sont dans l'unité du fichier ini. Ce changement a été fait car la position d'un point change selon que G20 ou G21 était actif lors de la programmation d'un G28, G30, G10 L2 ou G92.3.

### 21.1.3 Table d'outils, longueur et diamètre sont dans l'unité du fichier ini

Dans LinuxCNC, les longueurs d'outil (compensation) et les diamètres sont spécifiés seulement dans l'unité du fichier ini. Cela est nécessaire, car la longueur et le diamètre de l'outil changent selon que G20 ou G21 étaient actifs à l'initialisation des modes G43, G41, G42. Il était donc impossible de lancer un G-code avec des unités machines non natives, ceci même lorsque le G-code est simple et bien formé (débutant par G20 ou G21 et sans changement d'unité tout au long du programme) sans changer la table d'outils.

### 21.1.4 G84, G87 ne sont pas implémentés

G84 et G87 ne sont pour le moment pas implémentés. Ils le seront dans une version futur de LinuxCNC.

### 21.1.5 G28, G30 avec des mots d'axe

Lorsqu'un G28 ou un G30 est programmé avec seulement quelques mots d'axe présents, LinuxCNC déplace seulement les axes nommés. Ce comportement est commun aux contrôleurs de machine. Pour déplacer certains axes à un point intermédiaire, puis déplacer tous les axes à un point prédéfini, écrire deux lignes de G-code:

```
G0 X- Y-   (édplace les axes au point éintermdiaire)
G28        (édplace tous les axes au point ééprdfini)
```

## 21.2 Ajouts à RS274/NGC

Différences qui ne changent pas le déroulement des programmes en RS274/NGC.

### 21.2.1 Codes de filetage G33 et G76

Ces codes ne sont pas définis dans RS274/NGC.

### 21.2.2 G38.2

La pointe de touche n'est pas rétractée après un mouvement G38.2. Ce mouvement de retrait sera ajouté dans une version futur de LinuxCNC.

### 21.2.3 G38.3 à G38.5

Ces codes ne sont pas définis dans RS274/NGC.

### 21.2.4 Les O-codes

Ces codes ne sont pas définis dans RS274/NGC

### 21.2.5 M50 à M53 Correcteurs de vitesse

Ces codes ne sont pas définis dans RS274/NGC.

### 21.2.6 M61 à M66

Ces codes ne sont pas définis dans RS274/NGC.

### 21.2.7 G43, G43.1

**Longueurs d'outil négatives**
: Les spécifications RS274/NGC précisent "il est prévu que" toutes les longueurs d'outils soient positives. Cependant, G43 fonctionne avec des longueurs d'outil négatives.

**Outils de tournage**
: La compensation de longueur d'outil G43 peut compenser l'outil à la fois en X et en Z. Cette fonctionnalité est surtout utile sur les tours.

**Longueurs d'Outil dynamiques**
: LinuxCNC permet la spécification d'une longueur d'outil calculée par G43.1 I K.

### 21.2.8 G41.1, G42.1 Compensation dynamique

LinuxCNC permet dans le G-code, la spécification d'un diamètre d'outil et en mode tour, l'orientation est également spécifiée. Le format est G41.1/G42.1 D L, où D est le diamètre et L (si spécifié) est l'orientation de l'outil de tournage.

### 21.2.9 G43 sans le mot H

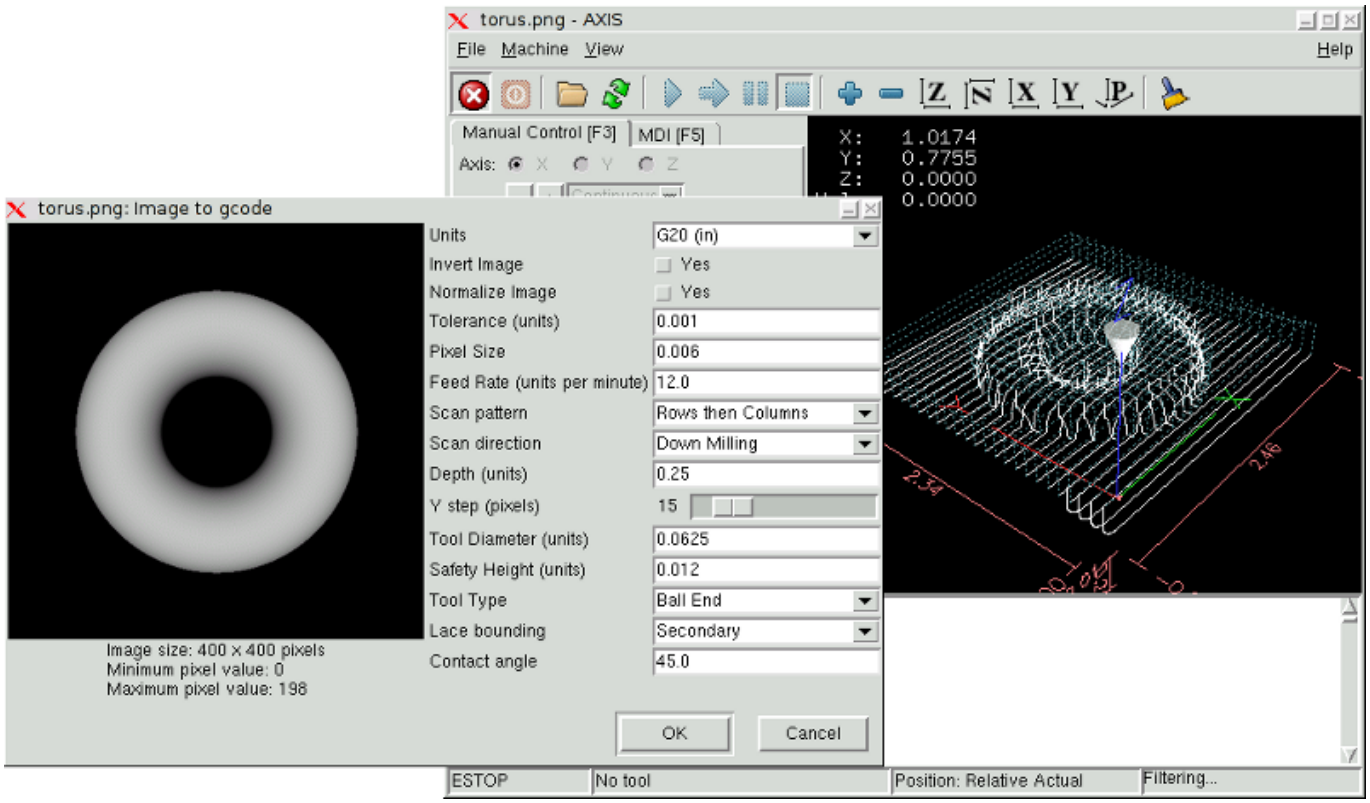
Ce code n'est pas permis en NGC. Dans LinuxCNC, il fixe la compensation de longueur pour l'outil actuellement chargé. Si aucun outil n'est actuellement chargé, c'est une erreur. Ceci a été fait afin que l'utilisateur n'ait pas à spécifier, pour chaque changement d'outil, le numéro d'outil à deux endroits et c'est cohérent avec la manière de fonctionner de G41/G42 quand le mot D n'est pas spécifié.

### 21.2.10 U, V et W axes

LinuxCNC peut admettre des machines ayant jusqu'à 9 axes en définissant un ensemble supplémentaire de 3 axes linéaires, connus comme U, V et W. :lang: fr :toc:

# Chapitre 22

# Image-to-gcode: Usiner un depth maps




## 22.1 Qu'est-ce qu'un *depth map*?

Il s'agit d'une image en échelle de gris dont la luminosité de chaque pixel corresponds à la profondeur (ou hauteur) de chaque point de l'objet.

## 22.2 Intégrer image-to-gcode dans l'interface utilisateur d'AXIS

Ajoutez les lignes suivantes dans la section: *[FILTER]* de votre fichier .ini pour qu'AXIS invoque automatiquement image-to-gcode à l'ouverture d'une image .png, .gif, ou .jpg:

```
PROGRAM_EXTENSION = .png,.gif,.jpg Grayscale Depth Image
```

Le fichier de configuration: *sim/axis.ini* est déjà configuré de cette façon.

## 22.3 Utilisation d'image-to-gcode

image-to-gcode peut être démarré soit en ouvrant une image dans AXIS, soit en invoquant image-to-gcode dans une console, de la manière suivante:

```
image-to-gcode torus.png > torus.ngc
```

Ajustez les réglages dans la colonne de droite, puis pressez *OK* pour créer le G-code. Selon la taille de l'image et les options choisies, le traitement peut durer de quelques secondes à quelques minutes. Quand une image est appelée, le G-code sera automatiquement chargé et visualisé dans AXIS une fois le traitement terminé. Dans AXIS, faites *Recharger* pour afficher de nouveau l'écran d'options d' image-to-gcode, vous pourrez ainsi travailler en boucle.

## 22.4 Les différentes options

### 22.4.1 Unités

Spécifie quelle unité sera utilisée dans le G-code généré G20 (pouces) ou G21 (mm), ce sera également l'unité utilisée par toutes les options marquées: *(units)*.

### 22.4.2 Invert Image

Si *no*, le pixel noir sera le point le plus bas et le pixel blanc le point le plus haut. Si *yes*, le pixel noir sera le point le plus haut et le pixel blanc le point le plus bas.

### 22.4.3 Normalize Image

Si *yes*, le pixel le plus sombre est ramené au noir, le pixel le plus lumineux est ramené au blanc.

### 22.4.4 Expand Image Border

Si *None*, l'image entrée sera utilisée telle-quelle, les détails les plus aux bords de l'image pourraient être coupés. Si *White* ou *Black*, alors une bordure de pixels égale au diamètre de l'outil sera ajoutée sur tout le pourtour pour éviter ce risque.

### 22.4.5 Tolerance (unités)

Quand une série de points est proche d'une ligne droite au point d'être dans la *tolerance* , elle sera traitée comme une ligne droite en sortie. Augmenter la tolérance peut donner de meilleures performances de contourage avec LinuxCNC, mais peut aussi estomper ou gommer les détails les plus fins de l'image.

### 22.4.6 Pixel Size (unités)

Il y a beaucoup d'unités pour un pixel dans l'image entrée. Habituellement ce nombre est beaucoup plus petit que 1.0. Par exemple, pour usiner un objet de 50x50mm depuis une image de 400x400 pixels, utiliser un *pixel size* de 0.125, parce que 50 / 400 = 0.125.

### 22.4.7 Plunge Feed Rate (unités par minute)

Vitesse du mouvement de plongée initial.

### 22.4.8 Feed Rate (unités par minute)

Vitesse d'avance pour le reste de l'usinage.

### 22.4.9 Spindle Speed (RPM)

Vitesse de rotation de la broche, en tours/mn

### 22.4.10 Scan Pattern

Modèles de balayage possibles:
– Rangées
– Colonnes
– Rangées puis colonnes
– Colonnes puis rangées

### 22.4.11 Scan Direction

Directions de balayage possibles:
– Positive: le fraisage commence à de petites valeurs de X ou Y et se poursuit avec des valeurs croissantes.
– Négative: le fraisage commence à des valeurs élevées de X ou Y et se poursuit avec des valeurs décroissantes.
– Alternative: le fraisage commence aux valeurs de X ou Y où s'est terminé le dernier mouvement. Cela réduit les déplacements *en l'air*.
– Up Milling: le fraisage commence en points bas et se poursuit vers les points hauts.
– Down Milling: le fraisage commence en points hauts et se poursuit vers les points bas.

### 22.4.12 Depth (unités)

Le dessus du bloc est toujours à *Z=0*. La profondeur d'usinage dans le matériau est de *Z=-depth.*

### 22.4.13 Step Over (pixels)

Distance entre rangées ou colonnes adjacentes. Pour trouver le nombre en pixels pour une distance donnée en unités, calculez: *distance/pixel size* et arrondissez au nombre le plus proche'.' Par exemple: si *pixel size=.006* et le pas souhaité sur la *distance=.015*, alors utilisez un Step Over de 2 ou 3 pixels, parce que *.015/.006=2.5*'.'

### 22.4.14 Tool Diameter

Le diamètre du taillant de l'outil.

### 22.4.15 Safety Height

La hauteur à laquelle les mouvements de traversée. image-to-gcode considère toujours le dessus du matériau comme étant: *Z=0*.

### 22.4.16 Tool Type

La forme du taillant de l'outil. Les formes possibles sont:
– Hémisphèrique
– Plate
– Vé à 45 degrés
– Vé à 60 degrés

### 22.4.17 Lace bounding

Contrôle si les zones relativement plates le long d'une colonne ou d'une rangée peuvent être ignorées. Ces options n'ont de sens que pour un fraisage dans les deux directions. Trois choix sont possibles:
- None: toutes les rangées et les colonnes seront entièrement fraisées.
- Secondary: lors du fraisage dans la deuxième direction, les zones qui ne présentent pas une forte pente dans cette direction seront ignorées.
- Full: lors du fraisage dans la première direction, les zones qui présentent une forte pente dans la deuxième direction seront ignorées. Lors du fraisage dans la deuxième direction, les zones qui ne présentent pas une forte pente dans cette direction seront ignorées.

### 22.4.18 Contact angle

Quand *Lace bounding* n'est pas None, les pentes qui présentent une pente supérieure à *Contact angle* seront considérées comme de *fortes* pentes et celles en dessous de cet angle considérées comme de faible pentes.

### 22.4.19 Offset d'ébauche et profondeur par passe d'ébauche

Image-to-gcode peut optionnellement produire des passes d'ébauche. La profondeur des passes d'ébauche successives est fixée par *Roughing depth per pass*. Par exemple, entrer 0.2 pour une première passe d'ébauche d'une profondeur de 0.2, la seconde passe d'ébauche aura une profondeur de 0.4 et ainsi de suite, jusqu'à-ce que la profondeur totale Depth de l'image soit atteinte. Aucune des passes d'ébauche n'usinera plus près de la partie finale que Roughing Offset. La figure ci-dessous montre une grande profondeur verticale à usiner. Sur cette image, la profondeur des passes d'ébauche est de 0.2 pouces et Roughing Offset de 0.1 pouces.

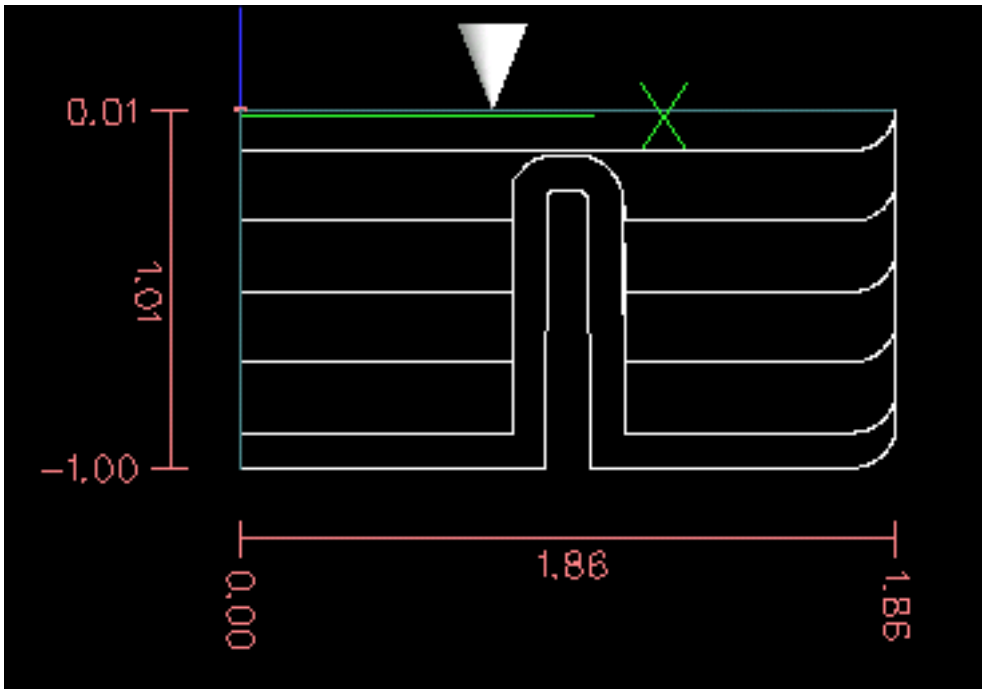



FIGURE 22.1 – Passes d'ébauche

# Chapitre 23

# Glossary

A listing of terms and what they mean. Some terms have a general meaning and several additional meanings for users, installers, and developers.

**Acme Screw**
A type of lead-screw that uses an Acme thread form. Acme threads have somewhat lower friction and wear than simple triangular threads, but ball-screws are lower yet. Most manual machine tools use acme lead-screws.

**Axis**
One of the computer controlled movable parts of the machine. For a typical vertical mill, the table is the X axis, the saddle is the Y axis, and the quill or knee is the Z axis. Angular axes like rotary tables are referred to as A, B, and C. Additional linear axes relative to the tool are called U, V, and W respectively.

**Axis(GUI)**
One of the Graphical User Interfaces available to users of LinuxCNC. It features the modern use of menus and mouse buttons while automating and hiding some of the more traditional LinuxCNC controls. It is the only open-source interface that displays the entire tool path as soon as a file is opened.

**Backlash**
The amount of "play" or lost motion that occurs when direction is reversed in a lead screw. or other mechanical motion driving system. It can result from nuts that are loose on leadscrews, slippage in belts, cable slack, "wind-up" in rotary couplings, and other places where the mechanical system is not "tight". Backlash will result in inaccurate motion, or in the case of motion caused by external forces (think cutting tool pulling on the work piece) the result can be broken cutting tools. This can happen because of the sudden increase in chip load on the cutter as the work piece is pulled across the backlash distance by the cutting tool.

**Backlash Compensation**
Any technique that attempts to reduce the effect of backlash without actually removing it from the mechanical system. This is typically done in software in the controller. This can correct the final resting place of the part in motion but fails to solve problems related to direction changes while in motion (think circular interpolation) and motion that is caused when external forces (think cutting tool pulling on the work piece) are the source of the motion.

**Ball Screw**
A type of lead-screw that uses small hardened steel balls between the nut and screw to reduce friction. Ball-screws have very low friction and backlash, but are usually quite expensive.

**Ball Nut**
A special nut designed for use with a ball-screw. It contains an internal passage to re-circulate the balls from one end of the screw to the other.

**CNC**
Computer Numerical Control. The general term used to refer to computer control of machinery. Instead of a human operator turning cranks to move a cutting tool, CNC uses a computer and motors to move the tool, based on a part program.

**Comp**
A tool used to build, compile and install LinuxCNC HAL components.

**Configuration(n)**
A directory containing a set of configuration files. Custom configurations are normally saved in the users home/LinuxCNC/configs directory. These files include LinuxCNC's traditional INI file and HAL files. A configuration may also contain several general files that describe tools, parameters, and NML connections.

**Configuration(v)**
The task of setting up LinuxCNC so that it matches the hardware on a machine tool.

**Coordinate Measuring Machine**
A Coordinate Measuring Machine is used to make many accurate measurements on parts. These machines can be used to create CAD data for parts where no drawings can be found, when a hand-made prototype needs to be digitized for moldmaking, or to check the accuracy of machined or molded parts.

**Display units**
The linear and angular units used for onscreen display.

**DRO**
A Digital Read Out is a system of position-measuring devices attached to the slides of a machine tool, which are connected to a numeric display showing the current location of the tool with respect to some reference position. DROs are very popular on hand-operated machine tools because they measure the true tool position without backlash, even if the machine has very loose Acme screws. Some DROs use linear quadrature encoders to pick up position information from the machine, and some use methods similar to a resolver which keeps rolling over.

**EDM**
EDM is a method of removing metal in hard or difficult to machine or tough metals, or where rotating tools would not be able to produce the desired shape in a cost-effective manner. An excellent example is rectangular punch dies, where sharp internal corners are desired. Milling operations can not give sharp internal corners with finite diameter tools. A *wire* EDM machine can make internal corners with a radius only slightly larger than the wire's radius. A *sinker* EDM can make internal corners with a radius only slightly larger than the radius on the corner of the sinking electrode.

**LinuxCNC**
The Enhanced Machine Controller. Initially a NIST project. LinuxCNC is able to run a wide range of motion devices.

**LinuxCNCIO**
The module within LinuxCNC that handles general purpose I/O, unrelated to the actual motion of the axes.

**LinuxCNCMOT**
The module within LinuxCNC that handles the actual motion of the cutting tool. It runs as a real-time program and directly controls the motors.

**Encoder**
A device to measure position. Usually a mechanical-optical device, which outputs a quadrature signal. The signal can be counted by special hardware, or directly by the parport with LinuxCNC.

**Feed**
Relatively slow, controlled motion of the tool used when making a cut.

**Feed rate**
The speed at which a cutting motion occurs. In auto or mdi mode, feed rate is commanded using an F word. F10 would mean ten machine units per minute.

**Feedback**
A method (e.g., quadrature encoder signals) by which LinuxCNC receives information about the position of motors

**Feedrate Override**
A manual, operator controlled change in the rate at which the tool moves while cutting. Often used to allow the operator to adjust for tools that are a little dull, or anything else that requires the feed rate to be "tweaked".

**Floating Point Number**
A number that has a decimal point. (12.300) In HAL it is known as float.

**G-Code**
The generic term used to refer to the most common part programming language. There are several dialects of G-code, LinuxCNC uses RS274/NGC.

**GUI**
Graphical User Interface.

**General**
A type of interface that allows communications between a computer and a human (in most cases) via the manipulation of icons and other elements (widgets) on a computer screen.

**LinuxCNC**
An application that presents a graphical screen to the machine operator allowing manipulation of the machine and the corresponding controlling program.

**HAL**
Hardware Abstraction Layer. At the highest level, it is simply a way to allow a number of building blocks to be loaded and interconnected to assemble a complex system. Many of the building blocks are drivers for hardware devices. However, HAL can do more than just configure hardware drivers.

**Home**
A specific location in the machine's work envelope that is used to make sure the computer and the actual machine both agree on the tool position.

**ini file**
A text file that contains most of the information that configures LinuxCNC for a particular machine

**Instance**
One can have an instance of a class or a particular object. The instance is the actual object created at runtime. In programmer jargon, the Lassie object is an instance of the Dog class.

**Joint Coordinates**
These specify the angles between the individual joints of the machine. See also Kinematics

**Jog**
Manually moving an axis of a machine. Jogging either moves the axis a fixed amount for each key-press, or moves the axis at a constant speed as long as you hold down the key. In manual mode, jog speed can be set from the graphical interface.

**kernel-space**
See real-time.

**Kinematics**
The position relationship between world coordinates and joint coordinates of a machine. There are two types of kinematics. Forward kinematics is used to calculate world coordinates from joint coordinates. Inverse kinematics is used for exactly the opposite purpose. Note that kinematics does not take into account, the forces, moments etc. on the machine. It is for positioning only.

**Lead-screw**
An screw that is rotated by a motor to move a table or other part of a machine. Lead-screws are usually either ball-screws or acme screws, although conventional triangular threaded screws may be used where accuracy and long life are not as important as low cost.

**Machine units**
The linear and angular units used for machine configuration. These units are specified and used in the ini file. HAL pins and parameters are also generally in machine units.

**MDI**
Manual Data Input. This is a mode of operation where the controller executes single lines of G-code as they are typed by the operator.

**NIST**
National Institute of Standards and Technology. An agency of the Department of Commerce in the United States.

**Offsets**
An arbitrary amount, added to the value of something to make it equal to some desired value. For example, gcode programs are often written around some convenient point, such as X0, Y0. Fixture offsets can be used to shift the actual execution point of that gcode program to properly fit the true location of the vise and jaws. Tool offsets can be used to shift the "uncorrected" length of a tool to equal that tool's actual length.

**Part Program**
A description of a part, in a language that the controller can understand. For LinuxCNC, that language is RS-274/NGC, commonly known as G-code.

**Program Units**
The linear and angular units used in a part program. The linear program units do not have to be the same as the linear machine units. See G20 and G21 for more information. The angular program units are always measured in degrees.

**Python**
General-purpose, very high-level programming language. Used in LinuxCNC for the Axis GUI, the Stepconf configuration tool, and several G-code programming scripts.

**Rapid**
Fast, possibly less precise motion of the tool, commonly used to move between cuts. If the tool meets the workpiece or the fixturing during a rapid, it is probably a bad thing!

**Rapid rate**
The speed at which a rapid motion occurs. In auto or mdi mode, rapid rate is usually the maximum speed of the machine. It is often desirable to limit the rapid rate when testing a g-code program for the first time.

**Real-time**
Software that is intended to meet very strict timing deadlines. Under Linux, in order to meet these requirements it is necessary to install RTAI or RTLINUX and build the software to run in those special environments. For this reason real-time software runs in kernel-space.

**RTAI**
Real Time Application Interface, see https://www.rtai.org/, one of two real-time extensions for Linux that LinuxCNC can use to achieve real-time performance.

**RTLINUX**
See http://www.rtlinux.org, one of two real-time extensions for Linux that LinuxCNC can use to achieve real-time performance.

**RTAPI**
A portable interface to real-time operating systems including RTAI and RTLINUX

**RS-274/NGC**
The formal name for the language used by LinuxCNC part programs.

**Servo Motor**
Generally, any motor that is used with error-sensing feedback to correct the position of an actuator. Also, a motor which is specially-designed to provide improved performance in such applications.

**Servo Loop**
A control loop used to control position or velocity of an motor equipped with a feedback device.

**Signed Integer**
A whole number that can have a positive or negative sign. In HAL it is known as s32. (A signed 32-bit integer has a usable range of -2,147,483,647 to +2,147,483,647.)

**Spindle**
The part of a machine tool that spins to do the cutting. On a mill or drill, the spindle holds the cutting tool. On a lathe, the spindle holds the workpiece.

**Spindle Speed Override**
A manual, operator controlled change in the rate at which the tool rotates while cutting. Often used to allow the operator to adjust for chatter caused by the cutter's teeth. Spindle Speed Override assumes that the LinuxCNC software has been configured to control spindle speed.

**Stepconf**
An LinuxCNC configuration wizard. It is able to handle many step-and-direction motion command based machines. It writes a full configuration after the user answers a few questions about the computer and machine that LinuxCNC is to run on.

**Stepper Motor**
A type of motor that turns in fixed steps. By counting steps, it is possible to determine how far the motor has turned. If the load exceeds the torque capability of the motor, it will skip one or more steps, causing position errors.

**TASK**
The module within LinuxCNC that coordinates the overall execution and interprets the part program.

**Tcl/Tk**
A scripting language and graphical widget toolkit with which several of LinuxCNCs GUIs and selection wizards were written.

**Traverse Move**
A move in a straight line from the start point to the end point.

**Units**

See "Machine Units", "Display Units", or "Program Units".

**Unsigned Integer**

A whole number that has no sign. In HAL it is known as u32. (An unsigned 32-bit integer has a usable range of zero to 4,294,967,296.)

**World Coordinates**

This is the absolute frame of reference. It gives coordinates in terms of a fixed reference frame that is attached to some point (generally the base) of the machine tool.

# Chapitre 24

# Legal Section

## 24.1 Copyright Terms

Copyright (c) 2000-2015 LinuxCNC.org

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

## 24.2 GNU Free Documentation License

GNU Free Documentation License Version 1.1, March 2000

Copyright © 2000 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

**0. PREAMBLE**

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

**1. APPLICABILITY AND DEFINITIONS**

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary

Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

**2. VERBATIM COPYING**

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

**3. COPYING IN QUANTITY**

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

**4. MODIFICATIONS**

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five). C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

**5. COMBINING DOCUMENTS**

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

**6. COLLECTIONS OF DOCUMENTS**

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

**7. AGGREGATION WITH INDEPENDENT WORKS**

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

**8. TRANSLATION**

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

**9. TERMINATION**

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

**10. FUTURE REVISIONS OF THIS LICENSE**

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

**ADDENDUM**: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

# Chapitre 25

# Index

_
.axisrc, 29
Éditeur externe, 29
Étendues du programme, 19
3 et 4, 181
7 et 8, 182

**Numbers**
2
    3 et 4, 181
6
    7 et 8, 182

**A**
A/U, 14, 47, 48
ABANDON, 11
acme screw, 199
Aperçu machine CNC, 76
Appel de fichier, 172
Arrêt d'urgence, 18
arrêt optionnel, 49, 77
Arrêts optionnels, 79
Arrosage, 22
arrosage, 50, 77, 78
Auto, 11, 48, 54, 57, 65
axes, 76
axes linéaires primaires, 76
axes linéaires secondaires, 76
Axes rotatifs, 76
AXIS, 13, 26
Axis, 13
axis, 199
AXIS avec un tour, 26

**B**
backlash, 199
backlash compensation, 199
ball nut, 199
ball screw, 199
Block Delete, 107
Boucles, 170
Bouton effacement de bloc, 77
break, 170
Broche, 22
broche, 48, 50, 77

**C**
call, 169
Changement D'Outil Manuel, 26
chargement, 79
CNC, 3, 53, 199
Commentaires, 118
Commentaires spéciaux, 29
comp, 199
Compensation d'outil, 88
Concepts pour l'utilisateur, 69
Conditionnel: if
    elseif
        else, 171
continue, 170
Contrôle de trajectoire continue avec tolérance, 143
Contrôle manuel, 15, 20
coordinate measuring machine, 200
coordonnées polaires, 115
correcteur de vitesse, 11, 48
Correcteur de vitesse broche, 23
correcteur vitesse, 59
correcteur vitesse broche, 48, 77
Correcteurs de vitesse, 15, 23
correcteurs vitesse, 77
Cycles de perçage, 146
Cycles de perçage G81-G89, 146

**D**
display units, 200
do, 170
Données manuelles, 15
DRO, 200

**E**
EDM, 200
effacement de bloc, 79
else, 171
elseif, 171
    else, 171
encoder, 200

endif, 171
endsub, 169
endwhile, 170
Enregistrement des mesures, 119
ESTOP, 11, 58
Expressions, 113

**F**
F: Réglage de la vitesse d'avance travail, 173
feed, 200
feed rate, 200
feedback, 200
feedrate override, 200
Fichier d'outils, 80
Fonctions, 114
Format de la table d'outils, 89

**G**
G-Code, 200
G-code, 106
G-Code bonnes pratiques, 121
G-codes modaux, 117
G0 Interpolation linéaire en vitesse rapide, 124
G1 Interpolation linéaire en vitesse travail, 125
G10 L1, 89
G10 L1 Ajustements dans la table d'outils, 133
G10 L10, 89
G10 L10 modifie les offsets d'outil dans la table d'outils, 134
G10 L11, 89
G10 L11 modifie les offsets d'outil dans la table d'outils, 135
G10 L2 Établissement de l'origine d'un système de coordonnées, 133
G10 L20 Établissement de l'origine d'un système de coordonnées, 135
G17 Plan XY, 136
G18 Plan XZ, 136
G19 Plan YZ, 136
G2 Interpolation circulaire sens horaire, 125
G20 Pouce, 136
G21 Millimètre, 136
G28, 136
G28.1, 136
G3 Interpolation circulaire anti-horaire, 125
G30, 136
G30.1, 136
G33 Mouvement avec broche synchronisée, 137
G33.1 Taraudage rigide, 138
G38.2 Palpeur, 139
G38.3 Palpeur, 139
G38.4 Palpeur, 139
G38.5 Palpeur, 139
G4 Temporisation, 130
G40 Révocation de la compensation de rayon, 140
G41 Compensation d'outil, 140
G41.1 Compensation dynamique, 140
G42 Compensation d'outil, 140
G42.1 Compensation dynamique, 140
G43 Activation de la compensation de longueur d'outil, 141
G43.1 Compensation dynamique de longueur d'outil, 141
G49 Révocation de compensation de longueur d'outil, 141
G5 Cubic spline, 130
G5.1 Quadratic spline, 131
G5.2 G5.3 NURBS Block, 131
G53 Mouvement en coordonnées absolues, 142
G55, 83
G61 Trajectoire exacte, 143
G61.1 Arrêt exact, 143
G7 Mode diamètre sur les tours, 133
G73 Cycle de perçage avec brise copeaux, 143
G76 Cycle de filetage multi-passe, 144
G8 Mode rayon sur les tours, 133
G80 Révocation des codes modaux, 148
G81 Cycle de perçage, 149
G81-G89
    Cycles de perçage, 146
G82 Cycle de perçage avec tempo, 152
G83 Cycle de perçage avec débourrage, 152
G84 Cycle de taraudage, 153
G85 Cycle d'alésage, 153
G86 Cycle d'alésage, 153
G87 Alésage inverse, 153
G88 Cycle d'alésage, 153
G89 Cycle d'alésage avec tempo, 154
G90 Mode de déplacement absolu, 155
G91 Mode de déplacement relatif, 155
G92 Décalages d'origine des systèmes de coordonnées, 156
G93
    G94
        G95: Choix des modes de vitesse, 157
G94
    G95: Choix des modes de vitesse, 157
G95: Choix des modes de vitesse, 157
G96
    G97: Vitesse de coupe constante
        Vitesse de coupe en tr/mn, 157
G97: Vitesse de coupe constante
    Vitesse de coupe en tr/mn, 157
G98
    G99 Retrait à la position initiale
        Retrait sur R, 157
G99 Retrait à la position initiale
    Retrait sur R, 157
Gouttelettes, 48
Groupes modaux, 117
GUI, 199, 200

**H**
HAL, 201
home, 201

**I**
if, 171

Indirection, 172
INI, 201
Instance, 201
Interraction vitesse, 79

**J**
jog, 201
joint coordinates, 201

**K**
KEYSTICK, 66
kinematics, 201

**L**
lancer, 49
Le G-code, 123
lead screw, 201
Les autres codes, 173
Les M-codes, 159
Les nombres, 108
Linux, 4
LinuxCNC, 200
LinuxCNC User Introduction, 3
LinuxCNCIO, 200
LinuxCNCMOT, 200
Log général, 119
loop, 202

**M**
M-codes définis par l'utilisateur M100-M199, 167
M-codes modaux, 118
M0 Pause dans le programme, 159
M1 Pause optionnelle dans le programme, 159
M100 à M199 M-codes définis par l'utilisateur, 167
M19 Orientation de la broche, 161
M2 Fin de programme, 160
M3 Broche en sens horaire, 160
M30 Fin de programme avec déchargement pièce, 160
M4 Broche en sens anti-horaire, 160
M48
M49 Autoriser/Inhiber les correcteurs de vitesse, 162
M49 Autoriser/Inhiber les correcteurs de vitesse, 162
M5 Arrêt de broche, 160
M50 Contrôle du correcteur de vitesse travail, 162
M51 Contrôle du correcteur de vitesse broche, 162
M52 Contrôle vitesse adaptative, 162
M53 Contrôle coupure vitesse, 162
M6 Appel d'outil, 160
M60 Pause pour déchargement pièce, 160
M61 Correction du numéro de l'outil courant, 163
M62 Contrôle un bit de sortie numérique, 163
M66 Contrôle d'entrée numerique et analogique, 163
M67 Contrôle de sortie analogique synchronisée avec un mouvement, 164
M68 Contrôle de Sortie analogique directe, 164
M7 Arrosage gouttelettes, 161
M70 Save Modal State, 165
M71 Invalidate Stored Modal State, 165
M72 Restore Modal State, 165
M73 Save and Autorestore Modal State, 166
M8 Arrosage fluide, 161
M9 Arrêt des arrosages, 161
machine units, 201
Manuel, 11, 48, 56, 64
Marche/Arret, 18
MDI, 11, 22, 48, 58, 201
MDI), 48
Messages, 119
Messages de débogage, 119
Mini, 53
mini, 53
mots, 107

**N**
NGCGUI, 31
NIST, 201
Numéro de ligne, 107

**O**
O-codes, 169
offsets, 201
Opérateurs binaires, 113
Opérations unaires, 114
OpenGL, 13
Ordre d'exécution, 120
Orientations des outils de tour, 180
Origine Machine, 47
Origine Piece, 48
Outils en positions 1
2
3 et 4, 181
Outils en positions 5
6
7 et 8, 182
ouvrir, 49

**P**
Panneau de contrôle virtuel, 29
Paramètres, 108
paramètres, 80
Paramètres de sous-programme, 110
Paramètres nommés, 111
Paramètres nommés prédéfinis, 111
Paramètres numérotés, 109
Paramètres système, 113
Parcours d'outil, 19
parcours outil, 54
part Program, 201
Particularités des tours, 180
pas a pas, 49
pause, 49
point contrôlé, 77
Position: Absolue, 15
Position: Actuelle, 15
Position: Commandée, 15

Position: Relative, 15
Précédence des opérateurs, 114
program units, 201
Programmation RS274/NGC, 192
Python, 13, 26

**R**
rapid, 202
rapid rate, 202
rapide, 124
real-time, 202
Repeat, 171
reprise, 49
Retrait sur R, 157
return, 169
RS274NGC, 202
RTAI, 202
RTAPI, 202
RTLINUX, 202

**S**
S: Réglage de la vitesse de rotation de la broche, 173
servo motor, 202
Sherline, 53
Signed Integer, 202
Sous-programmes, 169
spindle, 202
stepper motor, 202
sub, 169
Systèmes de coordonnées, 82

**T**
T: Choix de l'outil, 173
Table des index du G Code, 123
TASK, 202
Tcl, 46
tempo, 78
Tk, 13, 46, 202
TkLinuxCNC, 46
tklinuxcnc, 46
Tool Touch Off, 16
Toucher, 15, 88
Touchy GUI, 43
trace d'outil, 61
Trajectoire contrôlée, 143
Traverse Move, 202

**U**
unités, 78
units, 203
Unsigned Integer, 203

**V**
vérifier, 49
vitesse d'avance, 78
Vitesse de coupe en tr/mn, 157
Vitesse de jog, 23
vitesse de jog, 48
Vitesse maxi, 23
Vue générale du G-code de LinuxCNC, 106

**W**
while, 170
world coordinates, 203