Geo-Code Project Proposal

Carroll Loy (crl2131)

Introduction

With the advent of 3d machining and the availability of affordable routers, CNC machines have become quite common. The driving force behind a CNC machine is G-Code, a language that is very cumbersome and verbose. The language focuses on the mechanics of the machine and quite often, this verbosity obscures the meaning of the final program.

Motivation

Geo-Code is a programming language that focuses on describing the geometrical entities that should be machined. By allowing the user to focus on the logic, the program will be a self-documenting description of the target output.

The user will create the geometric entities that compose the target motions and push them on the container. The container collects all the motions that will be in the final output. The compiler output of a Geo-Code program will be the geometric motions written in the G-Code language.

Syntax

Primitives

- Bool: True or False
- Int: Integer type
- Double: floating point value

Reference System: All geometric entities are created relative to the currently active reference system.

- Global: keyword to signify the global reference system
- Local: keyword to signify the local scope reference system
- Identity: keyword to identify identity reference system (u = x-axis, v = y-axis, origin = 0, 0, 0)

Reference System Functions:All functions return a copy of the reference system (the original reference system is not modified).

- Local.Rotate(angle, x, y, z): Rotate the reference system by angle degrees around the vector <x,y,z>
- Local.Translate(x,y,z): Translate the origin of the reference system by vector <x,y,z>

Geometric Entities:All geometric entities that can be created natively. All entities are created with respect to the currently active reference system.

- Line(u, v): Start point at origin and end point at point (u,v).
- Polyline(u1, v1, u2, v2, ...): First line begins at origin and ends at u1, v1. Second line starts at u1, v1 and ends at u2, v2. ...
- Rectangle(u, v): Lower left point is the origin, upper right point is (u,v).
- Circle(r): center is the origin, radius is r.
- Arc(r, sa, ea): center is the origin, radius is r, the start point is sa degrees from the x-axis, and end point is ea degrees from the x-axis
- Char(a): a geometric representation of the character.

Geometric Entity Functions: All functions return a copy of the geometric entity (the original entity is not modified).

- Translate(u, v)
- Rotate(u, v)
- Scale(u, v)

Control Flow

- If:
  …
  else
  …
- for variable in Int1 to Int2:
  Begin at Int1 and iterate until reaching Int2.
- while (pred):
  Conitnue while predicate is true.

Mathematical Operators

+	a + b Addition
-	a – b Subtraction
*	a * b Multiplication
/	a / b Division

Sample Code

```
let container = Container()

# Create 5 concentric circles of increasing depth
for x in 1 to 5:
        Local = Global.Translate(0, 0, x)
        let circle = Circle(6 - x)
        container = container.Feed(circle, 25)


return container
```

Sample Output

```
G0 X5 Y0 Z0
G02 X5 Y0 I-5 J0 F25
G0 X4 Y0 Z0
G02 X4 Y0 I-4 J0 F25
G0 X3 Y0 Z0
G02 X3 Y0 I-3 J0 F25
G0 X2 Y0 Z0
G02 X2 Y0 I-2 J0 F25
G0 X1 Y0 Z0
G02 X1 Y0 I-1 J0 F25
```