Copilot for TurboCAD PPM

Version: 0.1

1) What does TurboCAD Copilot PPM do?

With TC 2025.1, a new Agent was added to Copilot that assists users with creating and asking for help related to Parametric Part Modeling (PPM) scripts. This Agent is activated by adding the prefix "PPM:" to questions. For example:

PPM: Create a script for a 90-degree conic approximated by a 20-point spline.

or:

PPM: Explain the tools used to create and edit scripts, and the basic components of a script.

2) How do I access TurboCAD Copilot PPM?

PPM is integrated into Copilot. Copilot is accessed by selecting the "Tools: Palettes: Copilot Palette…" menu option. Type your prompt with the "PPM:" prefix and click the Submit button to process a response. PPM is part of the Copilot Professional subscription.

3) Provide a better understanding how it works from a technical perspective.

The PPM Agent uses a reasoning LLM model that has access to multiple data bases related to Parametric Part Modeling scripts. The top agent is a manager that uses heuristics to activate and manage sub agents. For example, there is an agent that specializes in gathering live/current data from the PPM git hub, an agent that specializes in validated scripts, and one that for documentation.

4) Can the PPM Agent knowledge base be extended?

The PPM Agent uses a collection of fixed and dynamic assets to create the knowledge base. The fixed assets include documentation, FAQ, and a large collection of validated script examples. The fixed assets are tokenized and converted into a N-dimensional database accessed by the sub agents. The dynamic assets are extracted from the GitHub repository to create additional databases. These databases are needed to keep performance and effective content search that provide context to the AI reasoning models.

Users can contribute to the dynamic asset by creating context at the PPM repository located at:

https://github.com/tim-olson/PPM

Follow the instructions on the READMe regarding contributing. The Agent is trained and instructed to cite sources and provide attribution and credit. If the Agent modifies an existing script it will attribute the author as "PPM Assistant" and cite the sources used to create the final script.

Users typically can add example scripts and documentation that are immediately accessed by the PPM Agent inside TurboCAD Copilot.

The contents of the repository are guided under a MIT license.

5) Fast start: how to test in 10 minutes

Open Tools → Palettes → Copilot Palette… (Copilot Professional required).

Open Tools → Palettes → Parametric Part Script Editor Palette (Platinum).

In the Copilot palette text entry window, type "PPM: Create a 5-point star and extrude 5 units"

Copy the Code section from the response.

Switch to the Parametric Part Script Editor Palette

Paste into the editor, and Save as .ppm.

Press Run. If geometry appears, the script is syntactically valid.

Change a parameter value, Run again. Verify the output updates parametrically.

If using a custom function (e.g., Box(...) implemented in Macro/box.ppm), confirm the Macro folder placement and relative paths.

6) When will TurboCAD Copilot PPM be released?

This depends on beta testing and how much training can occur to fine tune the results. If not TC2025.1, then beta will extend through to TC2026.

7) Troubleshooting quick reference

Script runs but no geometry: Ensure at least one Output(...) and the listed objects are defined.

"Identifier not defined": Define every identifier before use; check typos and case.

No parametric change: Verify parameters are used in geometry expressions.

Unit mismatch/scale off: Add Units(1[in]) or Units(1[mm]) explicitly at top.

External symbol not found: Confirm path relative to Macro and mask in FolderList(...).

Booleans fail: Check solids overlap; avoid zero-thickness geometry.

8) Reporting issues

Report all bugs in the TurboCAD 2025 beta forum. One bug per post. Please copy/paste your exact question such that the issue can be repeated.

Examples

Example A — 3D from 2D via Thickness

Prompt: "Create a parametric box using Thickness(Rectangle, H) with Length/Width/Height parameters and constraints."

Expected: 3 params, rectangle at origin, thickness to make a box, one Output.

```
// Box via Thickness from a Rectangle
L = Parameter("Length", 4, LINEAR, Interval(0.1, 20));
W = Parameter("Width", 3, LINEAR, Interval(0.1, 20));
H = Parameter("Height", 1.5, LINEAR, Interval(0.1, 20));
Rect = Rectangle(L, W);
Box = Thickness(Rect, H);
Output(Box);
```

Example B — Custom function usage (Macro/box.ppm)

Prompt: "Use a custom Box(x0,y0,z0,x1,y1,z1) function kept in a Macro folder. Then fillet one edge."

Expected: Calls Box(...) like a built-in; then G3Fillet(...) on a mid-edge point; one Output.

```
// Requires Macro/box.ppm to be present
x = Parameter("size", 5, LINEAR, GreaterThan(0));
r1 = Parameter("r1", 0.5, LINEAR, GreaterThan(0));
b0 = Box(0, 0, 0, x, x, x);
b1 = G3Fillet(b0, Point(x/2, 0, 0), Array(r1, r1*2));
Output(b1);
```

Example C — PPM Fundamentals

Prompt: "Explain how a PPM script should structure Parameters and the Output operator. Give a minimal example."

Expected: An explanation of Parameter(...) with type + constraints, then an Output(...) listing defined objects.

Validation Script (minimal 2D)

```
// Simple rectangle with rotation (fundamentals demo)
H = Parameter("Height", 5, LINEAR, Interval(0, 100));
L = Parameter("Length", 10, LINEAR, Interval(0, 200));
A = Parameter("Angle", 0, ANGULAR, Interval(0, 360));
R1 = Rectangle(H, L);
R2 = RotateZ(R1, A);
Output(R2);
```

Example D — Sweep (profile along path)

Prompt: "Make a swept solid by rotating a 2D profile into the ZX plane and sweeping along a 2D path polyline."

Expected: Polyline for profile → RotateX(, 90) → path polyline → Sweep(profile, path).

```
// Sweep sample
L = Parameter("Length", 5, LINEAR, Interval(0.005, 1000));
W = Parameter("Width", 3, LINEAR, Interval(0.005, 1000));
H = Parameter("Height", 1, LINEAR, Interval(0.1, 3));
FR = Parameter("Fillet Radius", 0.3, LINEAR, Interval(0.001, 100));
profile2D = Polyline(Point(0,0), Point(0,H), Point(-FR,H), Point(-FR,0),
Point(0,0));
profileZX = RotateX(profile2D, 90, 0, 0);
path2D = Polyline(Point(0,0), Point(0,W), Fillet(FR), Point(L,W), Fillet(FR),
Point(L,0), Fillet(FR), Point(0,0), Fillet(FR));
solid = Sweep(profileZX, path2D);
Output(solid);
```

Example E — Boolean subtraction (sphere with hole)

Prompt: "Create a sphere with a through-hole using BooleanSubtract of a cylinder."

Expected: Sphere radius R, cylinder from thickened circle moved through sphere.

```
// Sphere with axial hole
R = Parameter("Radius", 8, LINEAR, Interval(0.001, 1000));
s = Sphere(R);
c = Circle(R/3);
cyl = Thickness(c, R*2);
cylZ = Move(cyl, 0, 0, -R);
res = BooleanSubtract(s, cylZ);
Output(res);
```

Example F — Text objects & styles

Prompt: "Make a 3D text plate: text with font/style, extruded via Thickness, centered at origin."

Expected: Text, TextFont, TextStyle, extrude via Thickness.

```
// Text plate
msg = Parameter("Text", "PPM", TEXT);
ht = Parameter("TextHeight", 5, LINEAR, Interval(0.5, 50));
ang = Parameter("Angle", 0, ANGULAR, Interval(0, 360));
Tfont = TextFont(0, ht, ang, "Arial");
Tstyle = TextStyle(CENTER, MIDDLE, BOLD);
T = Text(msg, Tfont, Tstyle);
T3 = Thickness(T, ht/10);
Output(T3);
```

Example G — Units & scale

Prompt: "Show how Units(1[in]) vs Units(1[mm]) affects scale in a simple cylinder."

Expected: Two versions or a parameter switch toggling units, with clear dimensional effect.

```
// Units demo – cylinder height equals 10 units in selected system
U = Parameter("UseInches", 1, CHECKBOX);
Output(IF(U, Units(1[in]), Units(1[mm])));
R = Parameter("Radius", 5, LINEAR, GreaterThan(0));
H = Parameter("HeightUnits", 10, LINEAR, GreaterThan(0));
C = Thickness(Circle(R), H);
Output(C);
```

Example H — External symbol (static)

Prompt: "Insert a static symbol from a .tcw file located in ..\..\..\Drawings relative to the Macro folder."

Expected: StaticSymbol(...) with relative path comment; one Output of the loaded symbol.

```
// Static symbol load example
// Note: paths are resolved relative to the Macro subfolder next to this ppm
DrawingName = Parameter("Drawing", "Drawing1",
Set(FolderList("..\\..\\..\\Drawings", "*.tcw")));
S = StaticSymbol("..\\..\\..\\Drawings\\" DrawingName ".tcw");
Output(S);
```

Appendix — PPM snippet library

A1 — Move copies

```
RB = Parameter("BaseRadius", 2, LINEAR, Interval(0.1, 10));
RT = Parameter("TopRadius", 0.5, LINEAR, Interval(0, 10));
H = Parameter("Height", 4, LINEAR, Interval(0.1, 20));
con1 = Cone(H, RB, RT);
cx = Parameter("CenterX", 5, LINEAR, Interval(-10, 10));
cy = Parameter("CenterY", 0, LINEAR, Interval(-10, 10));
```

```
cz = Parameter("CenterZ", 0, LINEAR, Interval(-10, 10));
count = Parameter("Copies", 2, LINEAR, Interval(1, 10));
con2 = Move(con1, cx, cy, cz, count);
Output(con1, con2);
```

A2 — 4-edge fillet on a box

```
L = Parameter("Length", 5, LINEAR);
W = Parameter("Width", 3, LINEAR);
H = Parameter("Height", 1, LINEAR);
R = Parameter("Radius", 0.5, LINEAR);
g0 = Box(0,0,0,L,W,H);
g1 = G3Fillet(g0,
  Array(Point(L/2,0,0), Point(0,W/2,0), Point(L/2,W,0), Point(L,W/2,0)),
  Array(R,R,R,R,R,R,R,R));
Output(g1);
```

A3 — RefPoint for insertion

```
W = Parameter("Width", 200, LINEAR);
H = Parameter("Height", 100, LINEAR);
rf = RefPoint(W/2, 0, 0);
plate = Thickness(Rectangle(W, H), 10);
Output(plate, rf);
```